# HPE Aruba Networking Virtual Intranet Access (VIA) Client Version 4.7 Security Target

ST Version: 1.0.1

Date: 09-July-2026

Prepared by:

TERON LABS

www.teronlabs.com

# Contents

1 Security Target Introduction ....6
1.1 Security Target Reference ....6
1.2 TOE Reference ....6
1.3 TOE Overview ....7
1.3.1 Intended Method of Use ....7
1.3.2 Major Security Features ....7
1.3.3 TOE Type ....8
1.3.4 Non-TOE Hardware, Software and Firmware ....8
1.4 TOE Description ....8
1.4.1 TOE Architecture ....8
1.4.2 Physical Scope ....9
1.4.3 Logical Boundary ....10
2 Conformance Claims ....12
2.1 Statement of Conformance Claims ....12
2.2 Conformance Claim Rationale ....12
2.2.1 TOE Type Consistency Rationale ....12
2.2.2 Security Problem Definition Consistency ....13
2.2.3 Security Objective Consistency ....13
2.2.4 Security Requirements Consistency ....13
2.3 Technical Decisions ....13
2.4 Conventions ....14
2.5 Acronyms ....16
3 Security Problem Definition ....17
3.1 Threats ....17
3.2 Assumptions ....19
4 Security Objectives ....20
4.1 Security Objectives for the TOE ....20
4.2 Security Objectives for the Operational Environment ....22
5 Security Requirements ....23
5.1 Extended Requirements ....23
5.2 TOE Security Functional Requirements ....23

5.2.1 Cryptographic Support (FCS) ... 25
5.2.2 User Data Protection (FDP) ... 29
5.2.3 Identification and Authentication (FIA) ... 30
5.2.4 Security Management (FMT) ... 31
5.2.5 Privacy (FPR) ... 32
5.2.6 Protection of the TSF (FPT) ... 32
5.2.7 Trusted Path/Channels (FTP) ... 33
5.3 TOE Security Assurance Requirements ... 34
6 TOE Summary Specification ... 35
6.1 Cryptographic Support ... 35
6.1.1 FCS_CKM.1/AK: Cryptographic Asymmetric Key Generation ... 37
6.1.2 FCS_CKM.1/SK: Cryptographic Symmetric Key Generation ... 37
6.1.3 FCS_CKM.1/VPN: Cryptographic Key Generation (IKE) ... 38
6.1.4 FCS_CKM.2: Cryptographic Key Establishment ... 38
6.1.5 FCS_CKM.1: Cryptographic Key Generation Services ... 38
6.1.6 FCS_CKM_EXT.2: Cryptographic Key Storage ... 38
6.1.7 FCS_CKM_EXT.4: Cryptographic Key Destruction ... 40
6.1.8 FCS_COP.1/SKC: Cryptographic Operation – Encryption/Decryption ... 40
6.1.9 FCS_COP.1/Hash: Cryptographic Operation - Hashing ... 40
6.1.10 FCS_COP.1/Sig: Cryptographic Operation - Signing ... 40
6.1.11 FCS_COP.1/KeyedHash: Cryptographic Operation - Keyed-Hash Message Authentication ... 41
6.1.12 FCS_IPSEC_EXT.1: IPsec ... 41
6.1.13 FCS_RBG_EXT.1: Random Bit Generation, FCS_RBG_EXT.2: Random Bit Generation from Application ... 42
6.1.14 FCS_STO_EXT.1 Storage of Credentials ... 43
6.2 User Data Protection (FDP) ... 43
6.2.1 FDP_DAR_EXT.1: Encryption of Sensitive Application Data ... 43
6.2.2 FDP_DEC_EXT.1: Access to Platform Resources ... 43
6.2.3 FDP_NET_EXT.1: Network Communications ... 43
6.2.4 FDP_RIP.2: Full Residual Information Protection ... 43
6.3 Identification and Authentication ... 44
6.3.1 FIA_X509_EXT.1: X.509 Certificate Validation ... 44
6.3.2 FIA_X509_EXT.2: X.509 Certificate Authentication ... 44
6.4 Security Management ... 44

6.4.1 FMT_CFG_EXT.1: Secure by Default Configuration ..... 44
6.4.2 FMT_MEC_EXT.1: Supported Configuration Mechanism ..... 45
6.4.3 FMT_SMF.1: Specification of Management Functions ..... 45
6.4.4 FMT_SMF.1/VPN: Specification of Management Functions (VPN) ..... 45
6.5 Privacy ..... 45
6.5.1 FPR_ANO_EXT.1: User Consent for Transmission of Personally Identifiable Information ..... 45
6.6 Protection of the TSF ..... 45
6.6.1 FPT_AEX_EXT.1: Anti-Exploitation Capabilities ..... 45
6.6.2 FPT_API_EXT.1: Use of Supported Services and APIs ..... 46
6.6.3 FPT_IDV_EXT.1: Software Identification and Versions ..... 46
6.6.4 FPT_LIB_EXT.1: Use of Third-Party Libraries ..... 46
6.6.5 FPT_TST_EXT.1/VPN(1): TSF Self-Test (VPN Client) (Windows and Linux) and FPT_TST_EXT.1/VPN(2): TSF Self-Test (VPN Client) (Android) ..... 46
6.6.6 FPT_TUD_EXT.1(1) (Android): Trusted Update, FPT_TUD_EXT.1(2) (Windows and Linux): Trusted Update FPT_TUD_EXT.2: Integrity for Installation and Update ..... 47
6.7 Trusted Path/Channels ..... 48
6.7.1 FTP_DIT_EXT.1: Protection of Data in Transit ..... 48
6.8 ALC_TSU_EXT.1: Timely Security Updates ..... 48
7 Protection Profile Claims ..... 49
8 TOE Summary Specification Rationale ..... 51
Appendix A: Windows Platform APIs ..... 53
Appendix B: Android Platform APIs ..... 55
Appendix C: Linux Platform APIs ..... 75

# List of Tables

Table 1 Physical Scope ........ 9
Table 2: Acronyms ........ 16
Table 3 Threats drawn from the Base-PP ........ 17
Table 4 Threats drawn from the PP-Module ........ 17
Table 5 Assumptions Drawn from the Base-PP ........ 19
Table 6 Assumptions Drawn from the PP-Module ........ 19
Table 7: Security Objectives for the TOE ........ 20
Table 8: Security Objectives for the Operational Environment ........ 22
Table 9: TOE Security Functional Components ........ 23
Table 10: Assurance Components ........ 34
Table 11: CAVP Certificate References ........ 35
Table 12: Key Purpose/Key Storage/Key Destruction ........ 38
Table 13: SFR Protection Profile Sources ........ 49
Table 14: Security Functions vs. Requirements Mapping ........ 51

# 1 Security Target Introduction

This section introduces the Target of Evaluation (TOE) and provides the Security Target (ST) and TOE identification, ST and TOE conformance claims, ST conventions, glossary and list of abbreviations.

The TOE is the HPE Aruba Networking Virtual Intranet Access (VIA) Client version 4.7. The TOE is a VPN client application that remote workers and mobile users can install on their computers or mobile devices to securely connect to their enterprise network from remote locations using IPsec.

The focus of this evaluation is on the TOE functionality supporting the claims in the Protection Profile for Application Software, version 1.4 ([PP_APP_v1.4]) and PP-Module for VPN Clients, version 2.4 ([MOD_VPNC_V2.4]). The security functionality specified in [PP_APP_v1.4] and [MOD_VPNC_V2.4] includes protection of communications between the TOE and external IT entities, ability to verify the source and integrity of updates to the TOE, and use of NIST-validated cryptographic mechanisms.

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

- TOE Description (Section 2)—provides an overview of the TOE and describes the physical and logical boundaries of the TOE
- Security Problem Definition (Section 3)—describes the threats and assumptions that define the security problem to be addressed by the TOE and its environment
- Security Objectives (Section 4)—describes the security objectives for the TOE and its operational environment necessary to counter the threats and satisfy the assumptions that define the security problem
- IT Security Requirements (Section 5)—specifies the security functional requirements (SFRs) and security assurance requirements (SARs) to be met by the TOE
- TOE Summary Specification (Section 6)—describes the security functions of the TOE and how they satisfy the SFRs
- Protection Profile Claims (Section 7)—provides rationale supporting the claims for conformance of the ST and the TOE to [PP_APP_v1.4] and [MOD_VPNC_V2.4]

TOE Summary Specification Rationale (Section 8)—provides mappings and rationale for the security problem definition, security objectives, security requirements, and security functions to justify their completeness, consistency, and suitability.

## 1.1 Security Target Reference

| | |
|---|---|
| ST Title | HPE Aruba Networking Virtual Intranet Access (VIA) Client Version 4.7 Security Target |
| ST Version | 1.0.1 |
| ST Date: | 9-July-2026 |

## 1.2 TOE Reference

| | |
|---|---|
| TOE Identification | HPE Aruba Networking Virtual Intranet Access (VIA) Client version 4.7 |
| TOE Developer | HPE Aruba Networking |
| Evaluation Sponsor | HPE Aruba Networking |

## 1.3 TOE Overview

The Target of Evaluation (TOE) is the HPE Aruba Networking Virtual Intranet Access (VIA) Client version 4.7. The TOE is a software application with IPsec VPN client capability.

### 1.3.1 Intended Method of Use

VIA is a part of the HPE Aruba Networking remote networks solution intended for teleworkers and mobile users. VIA detects the network environment (trusted and untrusted) of the user and connects the users to the enterprise network. A trusted network refers to a protected office network that allows users to access the corporate intranet directly. Untrusted networks are public Wi-Fi hotspots, such as airports, cafes, or home networks.

The VIA solution includes VIA Client and the Mobility Controller. The Mobility Controller running ArubaOS delivers network services that always on network with controller clustering, maximize Wi-Fi performance, ensure seamless roaming, and extends secure VPN access for Remote APs, Instant APs and VIA VPN clients. Note that the Mobility Controller is not part of the Common Criteria evaluation. The product relies on a platform-provided mechanism to establish HTTPS connectivity with the Mobility Controller to acquire VPN configuration settings. This is an environmental mechanism and is therefore not within the TOE boundary.

### 1.3.2 Major Security Features

With respect to the security functionality of the TOE, the TSF is limited to the relevant functionality that is defined in the claimed [PP_APP_v1.4] and [MOD_VPNC_V2.4]. The logical boundary of the TOE is summarized in section 2.3.2. However, the following general capabilities are considered to be within the scope of the TOE:

- **Cryptographic Support**: The TOE implements NIST CAVP approved cryptographic algorithms. The TOE protects communication between itself and an HPE Aruba Networking Mobility Controller over an unprotected network using IPsec and uses various mechanisms to protect credential data at rest.
- **User Data Protection**: The TOE leverages platform-provided functionality to encrypt sensitive data (as part of HTTPS) and implements secure VPN communications between the TOE and a VPN Gateway.
- **Identification and Authentication**: The TOE provides the ability to use, store, and protect X.509 certificates that are used for IPsec Virtual Private Network (VPN) connections. The TOE performs peer authentication using certificates.
- **Security Management**: The TOE and its IPsec VPN are fully configurable by a combination of functions provided directly by the TOE and those available to the associated VPN gateway.
- **Privacy**: The TOE does not transmit PII over a network.
- **Protection of the TSF**: The TOE performs self-tests that cover the TOE as well as the functions necessary to securely update the TOE. The TOE includes the use only documented platform APIs.
- **Trusted Path/Channel**: The TOE implements IPsec to establish a trusted channel to an external VPN gateway.

This ST focuses on the security functionality of application software and IPsec VPN capabilities of the TOE. The TOE provides secure remote network connectivity for Android, Linux, and Windows mobile devices and workstations. The TOE has two primary purposes:

- to provide secure corporate access to employee workstations and smartphones from anywhere
- to provide ease-of-use for the end users and network administrators

The IPsec VPN capabilities are the primary function of the TOE. IPsec is used by the TOE to protect communication between itself and an HPE Aruba Networking Mobility Controller over an unprotected network.

VIA can be downloaded directly from an HPE Aruba Networking Mobility Controller, pushed out using enterprise management tools, installed manually, or install the Android version from the Google Play Store. An HPE Aruba Networking Mobility Controller is required to terminate connections from a VIA client – VIA is not a general-purpose VPN client that works with third-party VPN gateways.

### 1.3.3 TOE Type

The TOE is a software application with IPsec VPN client capability implementing the security features required for strict conformance with the Base-PP and the PP-Module. The PP and the PP-Module are used in accordance with the PP-Configuration. The TOE runs on an end-user device and communicates with a gateway located on a Mobility Controller. The TOE is not in distributed form.

### 1.3.4 Non-TOE Hardware, Software and Firmware

The TOE is solely the IPsec VPN client. Yet, it does require external IT devices to be properly operated. Specifically, the TOE requires the following items in the network environment:

- An OCSP server is required for the TOE to be able to perform certificate validation.
- The Mobility Controller is required in the operational environment because the TOE uses it as its VPN gateway. The VIA client can work in the evaluated configuration with the Mobility Controller running ArubaOS 8.13 from 8.13.1 and later, but has only been tested with ArubaOS 8.13.1
- An EAP server (such as a RADIUS server) may or may not be used for authentication.

## 1.4 TOE Description

### 1.4.1 TOE Architecture

The TOE provides secure connectivity for users when accessing an enterprise or corporate resource (example: workstation, server) from an untrusted or trusted network connection. By default, the TOE automatically launches and establishes a remote connection when the user logs in to their system from an untrusted network.

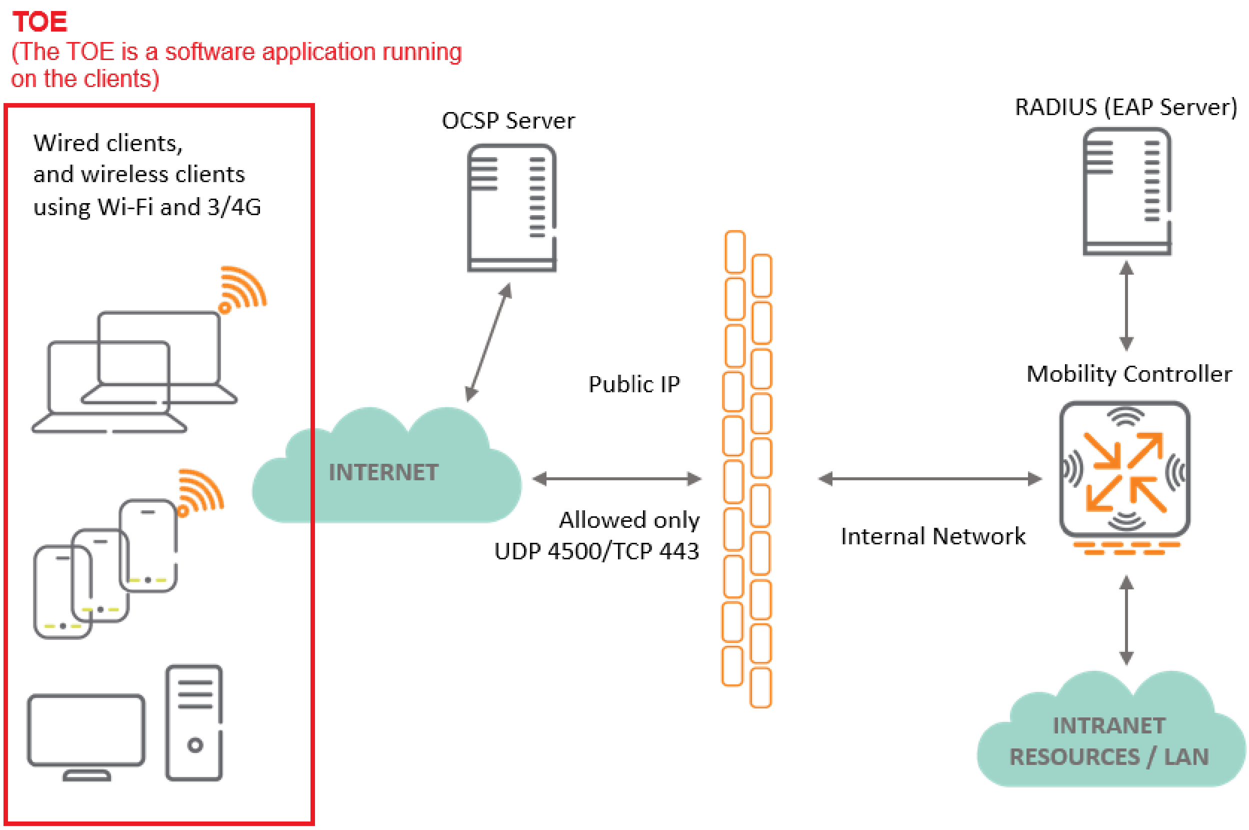
The TOE runs on an end-user device and communicates with a gateway located on a Mobility Controller. The server component is used to manage the client and ensure policies are enforced. The Mobility Controller maintains certain VIA configuration profiles, such as the VIA authentication profile, the VIA connection profile, and the VIA web authentication profile. Each profile plays an important role in authenticating the users and establishing a secure connection. When multiple authentication profiles are available, the VIA client prompts the user to select an authentication profile.

The first time a connection is established, a user opens the VIA client and enters the gateway name, username, and password. VIA then connects to the gateway over an HTTPS channel and attempts to authenticate using the user supplied credentials. The TSF relies on the underlying OS platform for establishment of the HTTPS channel; it is not part of the TOE itself. This connection is used to retrieve configuration settings for the IPsec connection and any site-specific branding (e.g. logo graphics).

If the VIA web authentication list has more than one VIA authentication profile, the user can choose a VIA authentication profile from the available ones. After successful authentication, the VIA client downloads the appropriate VIA connection profile and establishes the IPsec connection if the user is connected to an untrusted network.

At a protocol level, VIA operates over UDP port 4500, which is defined for IKE/IPsec traversal of NATs in RFC 3947. VIA uses HTTPS over TCP port 443 in order to contact the authentication server and download configuration profile updates before establishing each IKE/IPsec connection. The VIA topography is shown in the figure below.

*Figure 1: VIA Topography*




The OCSP server is required for the TOE to be able to perform certificate validation in accordance with the claim of FIA_X509_EXT.1 (see section 5.2.3.2). The Mobility Controller is required in the operational environment because the TOE uses it as its VPN gateway. The RADIUS (EAP Server) component is shown on the diagram because the Mobility Controller also functions as a wireless access system that uses EAP for client authentication. However, this functionality is separate from the VPN Client functionality and is outside the scope of the TOE and its operational environment; it is just included because a typical Mobility Controller deployment will have one present for its other functions.

### 1.4.2 Physical Scope

The TOE is the HPE Aruba Networking Virtual Intranet Access (VIA) Client version 4.7, a software application running on general-purpose personal computing and mobile device hardware. The TOE software, supported platforms, their software requirements, and tested hardware configurations are as follows:

*Table 1 Physical Scope*

| Platform | TOE Software | OS | Tested Hardware |
|---|---|---|---|
| Windows | Aruba-VIA-4.7.6.0.2603183-64.msi | Windows 11 64-bit[1] | HP Probook 440 G11 with Intel Core Ultra 7 155U |

[1] Windows clients using Secure Boot on a Windows Enterprise platform require the Long-Term Servicing Channel (LTSC) version of Windows 11

| | | | |
|---|---|---|---|
| Linux | cc-via_4.7.6.2603081-deb_amd64.deb | Ubuntu 24.04 | HP Probook 440 G11 with Intel Core Ultra 7 155U |
| Android | HPE-VIA-4.7.5.2603170.apk | Android 14 | Samsung Galaxy S24 |

Different platform versions of the TOE rely on their underlying OS platforms to varying extents. Where platform-specific differences exist within an individual SFR, the SFRs are iterated to indicate which claims apply to which platform. In all cases, the TSF relies on the OS platform to establish HTTPS connectivity to the VPN gateway for acquisition of IPsec configuration settings; this is outside the TOE boundary and therefore no SFRs apply to it.

The physical scope of the TOE also includes the following administrative guides:

- Aruba Virtual Intranet Access (VIA) 4.x Client Common Criteria Guidance, Version 2.5
- https://arubanetworking.hpe.com/techdocs/VIA/HPE-Aruba-VIA/Content/home.htm

### 1.4.3 Logical Boundary

This section summarizes the security functions provided by HPE Aruba Networking Virtual Intranet Access (VIA) client:

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

#### 1.4.3.1 Cryptographic Support

The TOE implements NIST CAVP approved cryptographic algorithms.

The TOE does not store keys unencrypted in persistent storage. While the TOE manipulates keys, on all platforms, the TOE platform's key storage is used. The TOE also uses various mechanisms to protect credential data at rest.

The IPsec implementation is the primary function of the TOE. IPsec is used by the TOE to protect communication between itself and an HPE Aruba Networking Mobility Controller over an unprotected network.

The TOE implements IKEv2 as defined in RFCs 7296 (with mandatory support for NAT traversal as specified in section 2.23), RFC 8784, RFC 8247 in tunnel mode only. The TOE does not offer transport mode as a configuration option. The TOE implements peer authentication using RSA certificates or ECDSA certificates that conform to RFC 4945. If certificates are used, the TOE ensures that the distinguished name (DN) contained in a certificate matches the expected DN for the entity attempting to establish a connection and ensures that the certificate has not been revoked (using the Online Certificate Status Protocol [OCSP] in accordance with RFC 6960).

The TOE implements the IPsec protocol as specified in RFC 4301; however, the TOE relies upon the VPN Gateway to ensure that the cryptographic algorithms and key sizes negotiated during the IKEv2 negotiation ensure that the security strength of the IKE_SA is greater than or equal to that of the CHILD_SA

The IKEv2 SA lifetimes are configured by the VPN Gateway based upon the length of time. Length of time includes 24 hours or less for Phase 1 and 8 hours or less for Phase 2.

#### 1.4.3.2 User Data Protection

The TOE uses the platform's permission mechanism to get a user's approval for access as part of the installation process. The TOE leverages platform-provided functionality to encrypt sensitive data and allows network communications to be initiated by the user in order to connect to the VPN Gateway. The TOE can also provide always-on functionality for application-initiated network communication.

The TOE ensures that residual information is protected from potential reuse in accessible objects such as network packets.

#### 1.4.3.3 Identification and Authentication

The TOE provides the ability to use, store, and protect X.509 certificates that are used for IPsec Virtual Private Network (VPN) connections. The TOE performs peer authentication using certificates.

#### 1.4.3.4 Security Management

The TOE and its IPsec VPN are fully configurable by a combination of functions provided directly by the TOE and those available to the associated VPN gateway. The TOE is not provided with any default. All external configuration comes from the Mobility Controller. The configuration options for the TOE consists of the URL of the gateway and the credentials used for the connection. The configuration options are stored and set using the mechanisms supported by the platform.

#### 1.4.3.5 Privacy

The TOE does not transmit PII over a network.

#### 1.4.3.6 Protection of the TSF

The TOE performs self-tests that cover the TOE as well as the functions necessary to securely update the TOE.

The TOE includes the use of only the documented platform APIs.

For each platform, 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.

HPE Aruba Networking provides a version control system for its software components.  The TOE has a unique software versioning that identifies major versions and their subsequent maintenance releases.

The About tab on the TOE displays the current system image version number for the TOE. The TOE platforms support loading updates by the administrator. For Windows and Linux platforms, the administrator obtains the update in the form of an installer through the HPE Aruba Networking Mobility Controller or the HPE Networking Support Portal. The update is verified using an RSA 2048 with SHA-256 digital signature. For Android versions, the application and signature are provided to and verified by the Google Play Store.

The TOE does not download, modify, replace or update its own binary code. The application is packaged such that its removal results in the deletion of all traces of the application, with the exception of configuration settings and output files.

#### 1.4.3.7 Trusted Path/Channel

TOE connects to the server over an HTTPS connection and attempts to authenticate using the user supplied credentials. The cryptography for the initial HTTPS connection is provided by the platform and is therefore outside the scope of the TOE. The IKE/IPsec transversal is secured using the TOE cryptography.

The TOE acts as a VPN client using IPsec to established secure channels to corresponding VPN gateways.

# 2 Conformance Claims

This section states the Conformance Claims for the ST and the TOE. This includes a statement of the Conformance Claims, a statement of the Conformance Claim Rationale, and the Identification of the Technical Decisions applicable to the TOE.

## 2.1 Statement of Conformance Claims

The ST and the TOE claim conformance to Common Criteria Version 3.1 Revision 5, Part 1 through to Part 3 identified in the following:

– Common Criteria for Information Technology Security Evaluation, Part 1: Introduction and general model, April 2017, Version 3.1 Revision 5, CCMB-2017-04-001

– Common Criteria for Information Technology Security Evaluation, Part 2: Security functional components, April 2017, Version 3.1 Revision 5, CCMB-2017-04-002

– Common Criteria for Information Technology Security Evaluation, Part 3: Security assurance components, April 2017, Version 3.1 Revision 5, CCMB-2017-04-003

The ST claims CC Part 2 conformance as CC Part 2 Extended.

The ST claims CC Part 3 conformance as CC Part 3 Extended.

The TOE does not claim conformance to any packages.

The ST claims conformance to the following Protection Profile, and the Protection Profile Module:

- Protection Profile for Application Software, Version 1.4, 2021-10-07 [PP_APP_v1.4]
- PP-Module for Virtual Private Network (VPN) Clients, Version 2.4, 2022-03-31 [MOD_VPNC_V2.4].

Conformance to the Base-PP and the PP-Module is claimed in accordance with the PP-Configuration:

- PP-Configuration for Application Software and Virtual Private Network (VPN) Clients, Version 1.3, 2023-04-07 [CFG-APP-VPNC_V1.3]

The ST claims no conformance to any Evaluation Assurance Level or any other security assurance requirement package. Instead, the security assurance requirements applicable to the TOE are those drawn from the Base-PP as required by Sect. 2.2 of CFG_APP-VPNC_V1.3.

The ST claims conformance to the Protection Profile for Application Software, Version 1.4, released 18-October-2021 (PP_APP_v1.4) as PP-conformant.

The ST claims conformance to the PP-Configuration for Application and Software and Virtual Private Network (VPN) Clients, Version 1.3, released 7-April-2023 (CFG-APP-VPNC_V1.3) as PP-configuration-conformant.

The ST claims exact conformance to the Base-PP, exact conformance to the PP-Module, and exact conformance to the PP-configuration.

## 2.2 Conformance Claim Rationale

### 2.2.1 TOE Type Consistency Rationale

The TOE is a software application with IPsec VPN client capability implementing the security features required for strict conformance with the Base-PP and the PP-Module. The PP and the PP-Module are used in accordance with the PP-Configuration. These are exactly the PP, the PP-Module, and the PP-configuration claimed in Sect. 2.1. The PP and PP-Module are exactly as identified in Sect. 1.3 of the PP-Configuration.

This ensures that the TOE Type is consistent with the TOE Type in the Base-PP, PP-Module, and PP-Configuration.

### 2.2.2 Security Problem Definition Consistency

The statement in the Security Problem Definition in this ST is reproduced exactly from the Base-PP and PP-Module. The resulting Security Objectives statement is a union of the Security Objectives of the Base-PP and the PP-Module. This ensures that the statement of the Security Objectives is consistent with the PP-Configuration.

### 2.2.3 Security Objective Consistency

The statement of the Security Objectives in this ST is reproduced exactly from the Base-PP and PP-Module. The resulting Security Objectives Statement is a union of the Security Objectives of the Base-PP and the PP-Module. This ensures that the statement of the Security Objectives is consistent with the PP-configuration.

### 2.2.4 Security Requirements Consistency

The security functional requirements are drawn exactly from the Base-PP and the PP-Module. The statement of the security functional requirements includes all mandatory and relevant selection-based requirements, and an optional requirement.

The security assurance requirements are drawn from the Base-PP only. This is consistent with Sect. 2.2 of the PP-Configuration. This ensures the consistency of the Security Assurance Requirements.

## 2.3 Technical Decisions

The following NIAP Technical Decisions for [PP_APP_v1.4] and [MOD_VPNC_V2.4] apply to the materials to which the TOE conforms, the TOE evidence, or how the TOE is evaluated:

- TD0964: Clarifications to FMT_MEC_EXT.1 Windows Test (supersedes TD0893)
- TD0711: FMT_SMF.1 direction when using MDF 3.3
- TD0945: Adding FIPS 186-5 in PP_APP_V1.4 (supersedes TD0860)
- TD0931: Clarification when CTR_DRBG is Selected for FCS_RBG_EXT.2.2 in PP_APP_V1.4
- TD0897: RFC 8784 Optional in VPNC 2.4
- TD0876: Updating FIPS 186-4 to 186-5 in MOD_VPNC_V2.4 (supersedes TD0697)
- TD0865: Consistency of Cryptographic Key Sizes
- TD0844: Addition of Assurance Package for Flaw Remediation V1.0 Conformance Claim
- TD0823: Update to Microsoft Windows Exploit Protection link in FPT_AEX_EXT.1.3
- TD0822: Correction to Windows Manifest File for FDP_DEC_EXT.1
- TD0815: Addition of Conditional TSS Activity for FPT_AEX_EXT.1.5
- TD0798: Static Memory Mapping Exceptions
- TD0788: Terminology Change in MOD_VPNC: Extended to Functional Package
- TD0780: FIA_X509_EXT.1 Test 4 Clarification

- TD0756: Update for platform-provided full disk encryption
- TD0753: MOD_VPNC FTP_DIT_EXT.1 Alignment for App PP 1.4
- TD0747: Configuration Storage Option for Android
- TD0743: FTP_DIT_EXT.1.1 Selection exclusivity
- TD0725: Correction to FCS_CKM_EXT.2/4 selections
- TD0719: ECD for PP APP V1.3 and 1.4
- TD0717: Format changes for PP_APP_V1.4
- TD0690: Missing EAs for FDP_VPN_EXT.1
- TD0672: VPN Client PP-Module updated to allow for new PP and PP-Module Versions
- TD0664: Testing activity for FPT_TUD_EXT.2.2
- TD0662: Changes to Testing IPsec NAT Transversal and XAUTH in MOD_VPNC 2.4
- TD0650: Conformance claim sections updated to allow for MOD_VPNC_V2.3 and 2.4
- TD0647: Table 2 Applicability
- TD0628: Addition of Container Image to Package Format

Technical Decisions not applicable to the evaluation:

- TD0736: Number of elements for iterations of FCS_HTTPS_EXT.1
  - The TD is not applicable since FCS_HTTPS_EXT.1 is not included in the evaluation.
- TD0711: FMT_SMF.1 direction when using MDF 3.3
  - The TD is not applicable since MDF 3.3 is not included in the evaluation.

## 2.4 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 either by adding a string starting with "/" (e.g. "FPT_TST_EXT.1/VPN") or by using a bracketed number (e.g. FCS_COP.1(1)).
  - 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.

## 2.5 Acronyms

*Table 2: Acronyms*

| Acronym | Definition |
|---|---|
| AES | Advanced Encryption Standard |
| API | Application Programming Interface |
| CBC | Cipher-Block Chaining |
| CA | Certificate Authority |
| CAVP | Cryptographic Algorithm Validation Program |
| CM | Configuration Management |
| CSP | Critical Security Parameter |
| DH | Diffie-Hellman |
| ECC | Elliptic-curve cryptography |
| FFC | Finite Field Cryptography |
| FIPS | Federal Information Processing Standard |
| HMAC | Hashed Message Authentication Code |
| HTTP | Hypertext Transfer Protocol |
| HTTPS | Hypertext Transfer Protocol Secure |
| IPsec | Internet Protocol Security |
| NIST | National Institute of Standards and Technology |
| OCSP | Online Certificate Status Protocol |
| RSA | Rivest, Shamir and Adleman (algorithm for public-key cryptography) |
| SAR | Security Assurance Requirement |
| SFR | Security Functional Requirement |
| SHA | Secure Hash Algorithm |
| SPD | Security Policy Database |
| ST | Security Target |
| TLS | Transport Layer Security |
| TOE | Target of Evaluation |
| TSF | TOE Security Functions |
| VIA | HPE Aruba Networking Virtual Intranet Access |

# 3 Security Problem Definition

This security target includes by reference the Security Problem Definition from the [PP_APP_v1.4] and [MOD_VPNC_V2.4]. 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] and [MOD_VPNC_V2.4] have presented a Security Problem Definition appropriate for application software that provides a Virtual Private Network used to establish a secure IPsec connection between a host platform and a remote system. As such, the [PP_APP_v1.4] and [MOD_VPNC_V2.4] Security Problem Definition applies to the TOE.

The following Threats and Assumptions are reproduced in this ST verbatim from the Base-PP and PP-Module. Both the Base-PP and PP-Module do not define any additional OSPs.

## 3.1 Threats

*Table 3 Threats drawn from the Base-PP*

| Threat ID | Threat Statement |
|---|---|
| T.NETWORK_ATTACK | An attacker is positioned on a communications channel or elsewhere on the network infrastructure. Attackers may engage in communications with the application software or alter communications between the application software and other endpoints in order to compromise it. |
| T.NETWORK_EAVESDROP | An attacker is positioned on a communications channel or else on the network infrastructure. Attackers may monitor and gain access to data exchanged between the application and other endpoints. |
| T.LOCAL_ATTACK | An attacker can act through unprivileged software on the same computing platform on which the application executes. Attackers may provide maliciously formatted input to the application in the form of files or other local communications. |
| T.PHYSICAL_ACCESS | An attacker may try to access sensitive data at rest. |

*Table 4 Threats drawn from the PP-Module*

| Threat ID | Threat Statement |
|---|---|
| T.UNAUTHORIZED_ACCESS | This PP-Module does not include requirements that can protect against an insider threat. Authorized users are not considered hostile or malicious and are trusted to follow appropriate guidance. Only authorized personnel should have access to the system or device that contains the IPsec VPN client. Therefore, the primary threat agents are the unauthorized entities that try to gain access to the protected network (in cases where tunnel mode is used) or to plaintext data that traverses the public network (regardless of whether transport mode or tunnel mode is used).<br><br>The endpoint of the network communication can be both geographically and logically distant from the TOE, and can pass through a variety of other systems. These intermediate systems may be under the control of the adversary, and offer an opportunity for communications over the network to be compromised. |

| | |
|---|---|
| | Plaintext communication over the network may allow critical data (such as passwords, configuration settings, and user data) to be read or manipulated directly by a malicious users or process on intermediate systems, leading to a compromise of the TOE or to the secured environmental systems that the TOE is being used to facilitate communications with. IPsec can be used to provide protection for this communication; however, there are numerous options that can be implemented for the protocol to be compliant to the protocol specification listed in the RFC. Some of these options can have negative impacts on the security of the connection. For instance, using a weak encryption algorithm (even one that is allowed by the RFC, such as DES) can allow an adversary to read and even manipulate the data on the encrypted channel, thus circumventing countermeasures in place to prevent such attacks. Further, if the protocol is implemented with little-used or non-standard options, it may be complaint with the protocol specification but will not be able to interact with other diverse equipment that is typically found in large enterprises.<br><br>Even though the communication path is protected, there is a possibility that the IPsec peer could be tricked into thinking a malicious third-party user or system is the TOE. For instance, a middleman could intercept a connection request to the TOE, and respond to the request as if it were the TOE. In a similar manner, the TOE could be tricked into thinking it is establishing communications with a legitimate IPsec peer when in fact it is not. An attacker could also mount a malicious man-in-the-middle-type of attack, in which an intermediate system is compromised, and the traffic is proxied, examined, and modified by this system. This attack can even be mounted via encrypted communication channels if appropriate countermeasures are not applied. These attacks are, in part, enabled by a malicious attack capturing network traffic (for instance, an authentication session) and “playing back” that traffic in order to fool an endpoint into thinking it was communicating with a legitimate remote entity. |
| T.TSF_CONFIGURATION | Configuring VPN tunnels is a complex and time-consuming process, and prone to errors if the interface for doing so is not well-specified or well-behaved. The inability or failure of an ignorant or careless administrator to configure certain aspects of the interface may also lead to the mis-specification of the desired communications policy of use of cryptography that mya be desired or required for a particular site. This may result in unintended weak or plaintext communications while the user thinks that their data are being protected. Other aspects of configuring the TOE or using its security mechanisms (for example, the update process) may also result in a reduction in the trustworthiness of the VPN client. |
| T.USER_DATA_REUSE | Data traversing the TOE could inadvertently be sent to a different user as a consequence of a poorly-designed TOE; since these data may be sensitive, this may cause a compromise that is unacceptable. The specific threat that must be addressed concerns user data that is retained by the TOE in the course of processing network traffic that could be inadvertently re-used in sending network traffic to a user other than that intended by the sender of the original network traffic. |
| T.TSF_FAILURE | Security mechanisms of the TOE generally build up from a primitive set of mechanisms (e.g., memory management, privileged modes of process execution) to more complex sets of mechanisms. Failure of the primitive mechanisms could lead to a compromise in more complex mechanisms, resulting in a compromise of the TSF. |

## 3.2 Assumptions

*Table 5 Assumptions Drawn from the Base-PP*

| Assumption ID | Assumption Statement |
|---|---|
| A.PLATFORM | The TOE relies upon a trustworthy computing platform with a reliable time clock for its execution. This includes the underlying platform and whatever runtime environment it provides to the TOE. |
| A.PROPER_USER | The user of the application software is not willfully negligent or hostile, and uses the software in compliance with the applied enterprise security policy. |
| A.PROPER_ADMIN | The administrator of the application software is not careless, willfully negligent or hostile, and administers the software in compliance with the applied enterprise security policy. |

*Table 6 Assumptions Drawn from the PP-Module*

| Assumption ID | Assumption Statement |
|---|---|
| A.NO_TOE_BYPASS | Information cannot flow on the network to which the VPN client's host is connected without passing through the TOE. |
| A.PHYSICAL | Physical security, commensurate with the value of the TOE and the data it contains, is assumed to be provided by the environment. |
| A.TRUSTED_CONFIG | Personnel configuring the TOE and its OE will follow the applicable security configuration guidance. |

# 4 Security Objectives

This ST is conformant to the Protection Profile for Application Software, Version 1.4, 2021-10-18 [PP_APP_v1.4] and PP-Module for Virtual Private Network (VPN) Clients, Version 2.4, 2022-03-31 [MOD_VPNC_V2.4].

This section reproduces only the corresponding Security Objectives for the TOE and the Security Objectives of the Operational Environment for reader convenience. The [PP_APP_v1.4] and [MOD_VPNC_V2.4] offer additional information about the identified security objectives but have not been reproduced here and the [PP_APP_v1.4] and [MOD_VPNC_V2.4] should be consulted if there is interest in that material.

In general, the [PP_APP_v1.4] and [MOD_VPNC_V2.4] have defined Security Objectives appropriate for a software application and IPsec VPN client and as such are applicable to the HPE Aruba Networking Virtual Intranet Access (VIA) Client TOE.

## 4.1 Security Objectives for the TOE

*Table 7: Security Objectives for the TOE*

| Objective | Description |
|---|---|
| [PP_APP_v1.4] | |
| O.INTEGRITY | Conformant TOEs ensure the integrity of their installation and update packages, and also leverage execution environment-based mitigations. Software is seldom, if ever, shipped without errors. The ability to deploy patches and updates to fielded software with integrity is critical to enterprise network security. Processor manufacturers, compiler developers, execution environment vendors, and operating system vendors have developed execution environment-based mitigations that increase the cost to attackers by adding complexity to the task of compromising systems.<br>Application software can often take advantage of these mechanisms by using APIs provided by the runtime environment or by enabling the mechanism through compiler or linker options. |
| O.QUALITY | To ensure quality of implementation, conformant TOEs leverage services and APIs provided by the runtime environment rather than implementing their own versions of these services and APIs. This is especially important for cryptographic services and other complex operations such as file and media parsing. Leveraging this platform behavior relies upon using only documented and supported APIs. |
| O.MANAGEMENT | To facilitate management by users and the enterprise, conformant TOEs provide consistent and supported interfaces for their security- relevant configuration and maintenance. This includes the deployment of applications and application updates through the use of platform-supported deployment mechanisms and formats, as well as providing mechanisms for configuration. This also includes providing control to the user regarding disclosure of any PII. |

| Objective | Description |
|---|---|
| O.PROTECTED_STORAGE | To address the issue of loss of confidentiality of user data in the event of loss of physical control of the storage medium, conformant TOEs will use data-at-rest protection. This involves encrypting data and keys stored by the TOE in order to prevent unauthorized access to this data. This also includes unnecessary network communications whose consequence may be the loss of data. |
| O.PROTECTED_COMMS | To address both passive (eavesdropping) and active (packet modification) network attack threats, conformant TOEs will use a trusted channel for sensitive data. Sensitive data includes cryptographic keys, passwords, and any other data specific to the application that should not be exposed outside of the application. |
| [MOD_VPNC_V2.4] | |
| O.AUTHENTICATION | To address the issues associated with unauthorized disclosure of information in transit, a compliant TOE's authentication ability (IPsec) will allow the TSF to establish VPN connectivity with a remote VPN gateway or peer and ensure that any such connection attempt is both authenticated and authorized. |
| O.CRYPTOGRAPHIC_FUNCTIONS | To address the issues associated with unauthorized disclosure of information in transit, a compliant TOE will implement cryptographic capabilities. These capabilities are intended to maintain confidentiality and allow for detection and modification of data that is transmitted outside of the TOE. |
| O.KNOWN_STATE | The TOE will provide sufficient measures to ensure it is operating in a known state. At minimum this includes management functionality to allow the security functionality to be configured and self-test functionality that allows it to assert its own integrity. It may also include auditing functionality that can be used to determine the operational behavior of the TOE. |
| O.NONDISCLOSURE | To address the issues associated with unauthorized disclosure of information at rest, a compliant TOE will ensure that non-persistent data is purged when no longer needed. The TSF may also implement measures to protect against the disclosure of stored cryptographic keys and data through implementation of protected storage and secure erasure methods. The TOE may optionally also enforce split-tunneling prevention to ensure that data in transit cannot be disclosed inadvertently outside of the IPsec tunnel and prohibit transmission of packets through a connection until certain conditions are met. |

## 4.2 Security Objectives for the Operational Environment

*Table 8: Security Objectives for the Operational Environment*

| Objective | Description |
|---|---|
| [PP_APP_v1.4] | |
| 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. |
| [MOD_VPNC_V2.4] | |
| OE.NO_TOE_BYPASS | Information cannot flow onto the network to which the VPN client's<br>host is connected without passing through the TOE. |
| OE.PHYSICAL | Physical security, commensurate with the value of the TOE and the data it contains, is assumed to be provided by the environment. |
| OE.TRUSTED_CONFIG | Personnel configuring the TOE and its OE will follow the applicable security configuration guidance. |

# 5 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 TOE and to scope the evaluation effort.

The SFRs have all been drawn from the [PP_APP_v1.4] and the [MOD_VPNC_V2.4].

As a result, any selection, assignment, or refinement operations already performed by that Protection Profile (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 TOE and ST claim the following additional requirements beyond the minimum that is required by the claimed PP-Configuration. Note per the claim of Exact Conformance that these additional requirements are drawn exclusively from those specified in the PP and PP-Module from the PP-Configuration as optional and selection-based requirements:

- From [PP_APP_V1.4]: FCS_CKM.1/AK, FCS_CKM.1/SK, FCS_CKM.2, FCS_COP.1(1), FCS_COP.1/Hash, FCS_COP.1/Sig, FCS_COP.1/KeyedHash, FCS_RBG_EXT.2, FIA_X509_EXT.1, FIA_X509_EXT.2, and FPT_TUD_EXT.2.

## 5.1 Extended Requirements

The ST references several extended components. Each one is taken verbatim from the Base-PP (as per TD0719) or the PP-Module. Only the operations allowed in the statement of the extended components are implemented in the ST. There are no additional or modified extended components included in the ST. Therefore, the statement of the extended components is exactly as in the Base-PP and the PP-Module. They are not repeated here.

## 5.2 TOE Security Functional Requirements

Table 9 identifies the SFRs satisfied by the TOE.

*Table 9: TOE Security Functional Components*

| Requirement Class | Requirement Component |
|---|---|
| **FCS: Cryptographic Support** | FCS_CKM.1/AK: Cryptographic Asymmetric Key Generation |
| | FCS_CKM.1/SK: Cryptographic Symmetric Key Generation |
| | FCS_CKM.1/VPN: Cryptographic Key Generation (IKE) |
| | FCS_CKM.1: Cryptographic Key Generation Services |
| | FCS_CKM.2: Cryptographic Key Establishment |
| | FCS_CKM_EXT.2: Cryptographic Key Storage |
| | FCS_CKM_EXT.4: Cryptographic Key Destruction |
| | FCS_COP.1(1): Cryptographic Operation – Encryption/Decryption |
| | FCS_COP.1/Hash: Cryptographic Operation - Hashing |

| Requirement Class | Requirement Component |
|---|---|
| | FCS_COP.1/Sig: Cryptographic Operation - Signing |
| | FCS_COP.1/KeyedHash: Cryptographic Operation - Keyed-Hash Message Authentication |
| | FCS_IPSEC_EXT.1: IPsec |
| | 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: 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 |
| | FDP_RIP.2: Full Residual Information Protection |
| **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 |
| | FMT_SMF.1/VPN: Specification of Management Functions (VPN) |
| **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_TST_EXT.1/VPN(1): TSF Self-Test (VPN Client) (Windows and Linux) |
| | FPT_TST_EXT.1/VPN(2): TSF Self-Test (VPN Client) (Android) |
| | FPT_TUD_EXT.1(1): Trusted Update (Android) |
| | FPT_TUD_EXT.1(2): Trusted Update (Windows and Linux) |
| | 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.1 Cryptographic Key Generation Services

**FCS_CKM.1.1** The application shall [

- ***implement asymmetric key generation***

].

### 5.2.1.2 FCS_CKM.1/AK Cryptographic Asymmetric Key Generation

**FCS_CKM.1.1/AK**[2] 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 [P-256]]*** that meet the following: *[**FIPS PUB 186-5, “Digital Signature Standard (DSS)”, Appendix B.4**]*; and,

[

- ***[FFC schemes] using Diffie-Hellman group 14 that meet the following: RFC 3526, Section 3***

].

### 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 [

- ***128 bit,***
- ***256 bit***

].

### 5.2.1.4 FCS_CKM.1/VPN Cryptographic Key Generation (IKE)

**FCS_CKM.1.1/VPN**[3] The TSF shall **[invoke platform-provided *functionality*] to** generate **asymmetric** cryptographic keys **used for IKE peer authentication** in accordance with: **[**

- ***FIPS PUB 186-5, “Digital Signature Standard (DSS)”, Appendix B.3 for RSA schemes;***
- ***FIPS PUB 186-5, “Digital Signature Standard (DSS)”, Appendix B.4 for ECDSA schemes and implementing “NIST curves”, P-256, P-384 and [no other curves]*]**

[2] TD0945, TD0876 and TD0717 applied.

[3] TD0876 applied.

and specified cryptographic key sizes [equivalent to, or greater than, a symmetric key strength of 112 bits].

#### 5.2.1.5 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"]****;* and

[

- ***[Key establishment scheme using Diffie-Hellman group 14]*** *that meets the following:* ***RFC 3526, Section 3***

].

#### 5.2.1.6 FCS_CKM_EXT.2 Cryptographic Key Storage

**FCS_CKM_EXT.2.1**[4] The [***TOE***] shall store persistent secrets and private keys when not in use in platform-provided key storage.

#### 5.2.1.7 FCS_CKM_EXT.4 Cryptographic Key Destruction

**FCS_CKM_EXT.4.1**[5] The [***TOE, TOE platform***] shall zeroize all plaintext secret and private cryptographic keys and CSPs when no longer required.

#### 5.2.1.8 FCS_COP.1/SKC Cryptographic Operation

**FCS_COP.1.1/SKC**[6] The **application** shall perform [*encryption/decryption*] in accordance with a specified cryptographic algorithm

- ***AES-CBC (as defined in NIST SP 800-38A) mode,***
- ***AES-GCM (as defined in NIST SP 800-38D)***

***mode***

**and** [

- ***no other modes***

**]** and cryptographic key sizes [***128-bit, 256-bit***].

#### 5.2.1.9 FCS_COP.1/Hash Cryptographic Operation - Hashing

**FCS_COP.1.1/Hash**[7] The **application** shall perform [*cryptographic hashing services*] in accordance with a specified cryptographic algorithm [

[4] Applied TD0725.
[5] Applied TD0725.
[6] Applied TD0717.
[7] Applied TD0717.

- ***SHA-1,***
- ***SHA-256,***
- ***SHA-384***

] and **message digest** sizes [

- ***160,***
- ***256,***
- ***384***

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

#### 5.2.1.10 FCS_COP.1/Sig Cryptographic Operation - Signing

**FCS_COP.1.1/Sig**[8] The **application** shall perform [cryptographic signature services *(generation and verification)*] in accordance with a specified cryptographic algorithm [

- ***RSA schemes using cryptographic key sizes of [2048-bit or greater] that meet the following: [FIPS PUB 186-5, "Digital Signature Standard (DSS)", Section 5],***
- ***ECDSA schemes using ["NIST curves" P-256, P-384 and [no other curves]] that meet the following: [FIPS PUB 186-5, "Digital Signature Standard (DSS)", Section 6]***

].

#### 5.2.1.11 FCS_COP.1/KeyedHash Cryptographic Operation - Keyed-Hash Message Authentication

**FCS_COP.1.1/KeyedHash**[9] The **application** shall perform [*keyed-hash message authentication*] in accordance with a specified cryptographic algorithm

- [***HMAC-SHA1, HMAC-SHA-256, HMAC-SHA-384***]

and [

- ***no other algorithms***

] **with** key sizes [**160, 256, 384 bits**] **and message digest sizes** [***256 , 384***] **and** [***160***] **bits** that meet the following: [***FIPS Pub 198-1*, *'The Keyed-Hash Message Authentication Code' and FIPS Pub 180-4 'Secure Hash Standard'***].

#### 5.2.1.12 FCS_IPSEC_EXT.1 IPsec

**FCS_IPSEC_EXT.1.1** The TSF shall implement the IPsec architecture as specified in RFC 4301.

**FCS_IPSEC_EXT.1.2** The TSF shall implement [***tunnel mode***].

**FCS_IPSEC_EXT.1.3** The TSF shall have a nominal, final entry in the SPD that matches anything that is otherwise unmatched, and discards it.

**FCS_IPSEC_EXT.1.4** The TSF shall implement the IPsec protocol ESP as defined by RFC 4303 using the cryptographic algorithms [AES-GCM-128, AES-GCM-256 as specified in RFC 4106,

[8] TD0717 applied.

[9] TD0717 applied.

**[AES-CBC-128, AES-CBC-256 (both specified by RFC 3602) together with a Secure Hash Algorithm (SHA)-based HMAC]**].

**FCS_IPSEC_EXT.1.5**[10] The TSF shall implement the protocol: [

- ***IKEv2 as defined in RFCs 7296 (with mandatory support for NAT traversal as specified in section 2.23), RFC 8247, and [no other RFCs for hash functions]***].

**FCS_IPSEC_EXT.1.6** The TSF shall ensure the encrypted payload in the [***IKEv2***] protocol uses the cryptographic algorithms *AES-CBC-128, AES-CBC-256 as specified in RFC 6379 and [**no other algorithm**]*].

**FCS_IPSEC_EXT.1.7** The TSF shall ensure that [

- ***IKEv2 SA lifetimes can be configured by [a VPN Gateway] based on [length of time],***

**].** If length of time is used, it must include at least one option that is 24 hours or less for Phase 1 SAs and 8 hours or less for Phase 2 SAs.

**FCS_IPSEC_EXT.1.8** The TSF shall ensure that all IKE protocols implement DH Groups

- **19 (256-bit Random ECP), 20 (384-bit Random ECP) according to RFC 5114 and**

[

- ***[14 (2048-bit MODP] according to RFC 3526***
- ***[no other DH Groups] according to RFC 5114***

].

**FCS_IPSEC_EXT.1.9** The TSF shall generate the secret value x used in the IKE DH key exchange (“x” in $g^x$ mod p) using the random bit generator specified in FCS_RBG_EXT.1, and having a length of at least [**224, 256, 384**] bits.

**FCS_IPSEC_EXT.1.10** The TSF shall generate nonces used in IKE exchanges in a manner such that the probability that a specific nonce value will be repeated during the life a specific IPsec SA is less than 1 in 2^[**112, 128, 192**].

**Application Note**: The probability that a specific nonce value will be repeated during the life of a specific IPsec SA is less than 1 in 2^112 for DH Group 14, 2^128 for DH Group 19, or 2^192 for DH Group 20.

**FCS_IPSEC_EXT.1.11** The TSF shall ensure that all IKE protocols perform peer authentication using [***RSA, ECDSA***] that use X.509v3 certificates that conform to RFC 4945 and [***no other method***].

**FCS_IPSEC_EXT.1.12** The TSF shall not establish an SA if the [***[Distinguished Name (DN)] and [no other reference identifier type]***] contained in a certificate does not match the expected value(s) for the entity attempting to establish a connection.

**FCS_IPSEC_EXT.1.13** The TSF shall not establish an SA if the presented identifier does not match the configured reference identifier of the peer.

[10] TD0897 applied.

**FCS_IPSEC_EXT.1.14** The [***VPN Gateway***] shall be able to ensure by default that the strength of the symmetric algorithm (in terms of the number of bits in the key) negotiated to protect the [***IKEv2 IKE_SA***] connection is greater than or equal to the strength of the symmetric algorithm (in terms of the number of bits in the key) negotiated to protect the [***IKEv2 CHILD_SA***] connection.

#### 5.2.1.13 FCS_RBG_EXT.1 Random Bit Generation Services

**FCS_RBG_EXT.1.1** The application shall [

- ***implement DRBG functionality***

] for its cryptographic operations.

#### 5.2.1.14 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**[11] 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 [

- ***384 bits***

] 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.

#### 5.2.1.15 FCS_STO_EXT.1(1) Storage of Credentials for Windows and Android

**FCS_STO_EXT.1.1(1)**[12] The application shall [

- ***invoke the functionality provided by the platform to securely store [RSA/ECDSA Certificates (IKEv2)],***

] to non-volatile memory.

#### 5.2.1.16 FCS_STO_EXT.1(2) Storage of Credentials for Linux

**FCS_STO_EXT.1.1(2)**[13] The application shall [

- ***implement functionality to securely store [RSA/ECDSA Certificates (IKEv2)] according to FCS_COP.1/SKC],***

] to non-volatile memory.

### 5.2.2 User Data Protection (FDP)

[11] TD0931 applied.
[12] TD0865 applied.
[13] TD0865 applied.

#### 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***

].

**FDP_DEC_EXT.1.2** The application shall restrict its access to [

- ***no sensitive information repositories***

].

#### 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** [**connection to the VPN Gateway**],*

].

#### 5.2.2.4 FDP_RIP.2 Full Residual Information Protection

**FDP_RIP.2.1** The [***TOE***] shall enforce that any previous information content of a resource is made unavailable upon the [***allocation of the resource to***] all objects.

### 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 [***implement functionality***] to validate certificates in accordance with the following rules:

- RFC 5280 certificate validation and certificate path validation.
- The certificate path must terminate with a trusted CA certificate.
- The application shall validate a certificate path by ensuring the presence of the basicConstraints extension, that the CA flag is set to TRUE for all CA certificates, and that any path constraints are met.
- The application shall validate that any CA certificate includes caSigning purpose in the key usage field
- The application shall validate the revocation status of the certificate using [***OCSP as specified in RFC 6960***]
- The application shall validate the extendedKeyUsage (EKU) field according to the following rules:
  - o 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.

**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**[14] The application shall use X.509v3 certificates as defined by RFC 5280 to support authentication for **IPsec** and [**no other protocols**].

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

### 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 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 [

- ***[management functions defined in FMT_SMF.1/VPN]***].

#### 5.2.4.4 FMT_SMF.1/VPN Specification of Management Functions (VPN)

**FMT_SMF.1.1/VPN** The TSF shall be capable of performing the following management functions:

[

- ***Specify VPN gateways to use for connections,***
- ***Specify client credentials to be used for connections***

[14] TD0788 applied.

]

### 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 [*[HPE internal versioning scheme]*].

#### 5.2.6.4 FPT_LIB_EXT.1(1) Use of Third-Party Libraries (Windows)

**FPT_LIB_EXT.1.1(1)** The application shall be packaged with only [**openssl (libeay32.dll, ssleay32.dll), System.Windows.Interactivity, PageAnimator.dll**].

#### 5.2.6.5 FPT_LIB_EXT.1(2) Use of Third-Party Libraries (Linux)

**FPT_LIB_EXT.1.1(2)** The application shall be packaged with only [**libanmocana.so, opensc-pkcs11.so, pkcs11-engine.so**].

#### 5.2.6.6 FPT_LIB_EXT.1(3) Use of Third-Party Libraries (Android)

**FPT_LIB_EXT.1.1(3)** The application shall be packaged with only [**no third-party library**].

#### 5.2.6.7 FPT_TST_EXT.1/VPN(1) TSF Self-Test (Windows and Linux)

**FPT_TST_EXT.1.1/VPN(1)** The [***TOE***] shall run a suite of self-tests during initial start-up (on power on) to demonstrate the correct operation of the TSF.

**FPT_TST_EXT.1.2/VPN(1)** The [***TOE***] shall provide the capability to verify the integrity of stored TSF executable code when it is loaded for execution through the use of the [**TOE cryptographic known-answer tests**].

#### 5.2.6.8 FPT_TST_EXT.1/VPN(2) TSF Self-Test (Android)

**FPT_TST_EXT.1.1/VPN(2)** The [***TOE***] shall run a suite of self-tests during initial start-up (on power on) to demonstrate the correct operation of the TSF.

**FPT_TST_EXT.1.2/VPN(2)** The [***TOE platform***] shall provide the capability to verify the integrity of stored TSF executable code when it is loaded for execution through the use of the [**TOE cryptographic known-answer tests during startup of the cryptographic module for Windows and Linux, integrity tests on the entire application at startup for Android**].

#### 5.2.6.9 FPT_TUD_EXT.1(1) Integrity for Installation and Update (Android)

**FPT_TUD_EXT.1.1(1)** The application shall [***leverage the platform***] to check for updates and patches to the application software.

**FPT_TUD_EXT.1.2(1)** The application shall [***provide the ability***] to query the current version of the application software.

**FPT_TUD_EXT.1.3(1)** The application shall not download, modify, replace or update its own binary code.

**FPT_TUD_EXT.1.4(1)** Application updates shall be digitally signed such that the application platform can cryptographically verify them prior to installation.

**FPT_TUD_EXT.1.5(1)** The application is distributed [*as an additional software package to the platform OS*].

#### 5.2.6.10 FPT_TUD_EXT.1(2) Integrity for Installation and Update (Windows and Linux)

**FPT_TUD_EXT.1.1(2)** The application shall [***provide the ability***] to check for updates and patches to the application software.

**FPT_TUD_EXT.1.2(2)** The application shall [***provide the ability***] to query the current version of the application software.

**FPT_TUD_EXT.1.3(2)** The application shall not download, modify, replace or update its own binary code.

**FPT_TUD_EXT.1.4(2)** Application updates shall be digitally signed such that the application platform can cryptographically verify them prior to installation.

**FPT_TUD_EXT.1.5(2)** The application is distributed [*as an additional software package to the platform OS*].

#### 5.2.6.11 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

**FTP_DIT_EXT.1.1**[15] The application shall encrypt all transmitted [sensitive data] using IPsec as specified in FCS_IPSEC_EXT.1 and **[*no other protocols*]** between itself and another trusted IT product.

## 5.3 TOE Security Assurance Requirements

The security assurance requirements for the TOE are included by reference according to TD0600: Conformance claim sections updated to allow for MOD_VPNC_V2.4. These SARs are consistent with those specified in PP-Configuration for Application Software and Virtual Private Network (VPN) Clients, Version 1.0, 2021-08-13.

*Table 10: 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
- ASE_REQ.1 Derived security requirements
- ASE_TSS.1 TOE summary specification
- ASE_SPD.1 Security Problem Definition

Consequently, the evaluation activities specified in PP-Configuration for Application Software and Virtual Private Network (VPN) Clients, Version 1.3, 2023-04-07 apply to the TOE evaluation.

[15] Applied TD0753 and TD0743.

# 6 TOE Summary Specification

This chapter describes the following security functions:

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

## 6.1 Cryptographic Support

For Windows and Android, the TSF relies on the OS platforms for AES encryption of credential data as well as full-stack implementation of HTTPS for initial connectivity to the VPN gateway to acquire configuration settings for IPsec. For Linux, the TSF implements AES encryption of credential data and relies on the OS platform for the full-stack implementation of HTTPS for initial connectivity to the VPN gateway to acquire configuration settings for IPsec.

The following table lists the CAVP certificates obtained for the cryptographic algorithms claimed in this ST. It should be noted that the testing requirements for FIPS PUB 186-4 and FIPS PUB 186-5 are identical for the algorithms relevant to this ST. Therefore, although the CAVP certificates listed in the table reference FIPS PUB 186-4, the claims for these algorithms are equally applicable to FIPS PUB 186-5.

*Table 11: CAVP Certificate References*

| Requirements | Functions | CAVP Certificates | | |
|---|---|---|---|---|
| | | Windows | Android | Linux |
| Cryptographic Asymmetric Key Generation | | | | |
| FCS_CKM.1/AK | ECC schemes using “NIST curves” P- 256, P-384 that meet the following: FIPS PUB 186-5, “Digital Signature Standard (DSS)”, Appendix B.4 | A6942 | A6942 | A6942 |
| | FFC schemes] using Diffie-Hellman group 14 that meet the following: RFC 3526, Section 3 | A6942 | A6942 | A6942 |
| Cryptographic Key Generation (IKE) | | | | |
| FCS_CKM.1.1/VPN | FIPS PUB 186-5, “Digital Signature Standard (DSS)”, Appendix B.3 for RSA schemes; | A6942 | A6942 | A6942 |
| | FIPS PUB 186-5, “Digital Signature Standard (DSS)”, Appendix B.4 for ECDSA schemes and implementing “NIST curves”, P-256 and P-384 | A6942 | A6942 | A6942 |

| | | | | |
|---|---|---|---|---|
| Cryptographic Key Establishment | | | | |
| FCS_CKM.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” | A6942 | A6942 | A6942 |
| | Key establishment scheme using Diffie- Hellman group 14 that meets the following: RFC 3526, Section 3 | A6942 | A6942 | A6942 |
| Cryptographic Operation – Encryption/Decryption | | | | |
| FCS_COP.1.1(1) | AES-CBC (128-bit, 256-bit) as defined in NIST SP 800-38A<br>AES-GCM (128-bit, 256-bit) as defined in NIST SP 800-38D | A6942 | A6942 | A6942 |
| Cryptographic Operation - Hashing | | | | |
| FCS_COP.1.1/Hash | SHA-1, SHA-256, and SHA-384 as FIPS Pub 180-4 | A6942 | A6942 | A6942 |
| Cryptographic Operation - Signing | | | | |
| FCS_COP.1.1/Sig | RSA schemes using cryptographic key sizes of 2048-bit or greater that meet the following: FIPS PUB 186-5, “Digital Signature Standard (DSS)”, Section 4 | A6942 | A6942 | A6942 |

| Requirements | Functions | Certificate | | |
|---|---|---|---|---|
| | | Windows | Android | Linux |
| | ECDSA schemes using “NIST curves” P- 256, P-384 that meet FIPS PUB 186-5, “Digital Signature Standard (DSS)”, Section 5 | A6942 | A6942 | A6942 |
| Cryptographic Operation - Keyed-Hash Message Authentication | | | | |
| FCS_COP.1.1/KeyedHash | HMAC-SHA-1<br>HMAC-SHA-256 HMAC-SHA-384<br>That meets the following: FIPS Pub 198-1 and FIPS Pub 180-4 | A6942 | A6942 | A6942 |
| Random Bit Generation Services<br>Random Bit Generation from Application | | | | |
| FCS_RBG_EXT.1<br>FCS_RBG_EXT.2 | 256-bits<br>NIST Special Publication 800-90A using CTR_DRBG (AES) | A6942 | A6942 | A6942 |

### 6.1.1 FCS_CKM.1/AK: Cryptographic Asymmetric Key Generation

The TOE generates asymmetric keys using the following key generation algorithms:

- Elliptic-curve cryptography (ECC) using NIST curves P-256 (256-bits) and P-384 (384-bits) key pairs for key establishment that meet FIPS PUB 186-5, “Digital Signature Standard (DSS)”, Appendix B.4 for IPsec (DH groups 19 and 20, respectively).
- Finite Field Cryptography (FFC) using Diffie-Hellman group 14 (2048-bits) that meets RFC 3526, Section 3 for IPsec.

The TOE invokes cryptographic key generation using the Aruba Common Cryptographic Module (ACCM).

### 6.1.2 FCS_CKM.1/SK: Cryptographic Symmetric Key Generation

The TOE generates symmetric cryptographic keys using a Random Bit Generator as specified in FCS_RBG_EXT.1 with 128 bit and 256 bit cryptographic key sizes.

The TOE includes an AES-256 CTR_DRBG (irrespective of the underlying Platform) that seeds itself with 384-bits of entropy (composed of a 256-bit entropy input and a 128-bit nonce) drawn from a Platform function intended to provide cryptographically secure randomness. The following list describes the mechanism by which VIA obtains its seeding.

- Android- /dev/random
- Windows - BCryptGenRandom()
- Linux - getrandom() and that it uses a flag of GRND_RANDOM and behaves exactly as /dev/random.

Based on NIAP’s “Clarification to the Entropy Documentation and Assessment Annex”, HPE Aruba Networking assumes a minimum entropy of 0.67 bits of entropy per bit of data from platform provided sources. This minimum-entropy estimate along with knowledge that the seed is 384-bits means that the TOE seeds itself with at least 256-bits of entropy.

### 6.1.3 FCS_CKM.1/VPN: Cryptographic Key Generation (IKE)

The TOE supports RSA (2048 bits) and ECDSA (for curves P-256 and P-384) key pairs for use with IKE peer authentication. The TOE delegates generation of the asymmetric keys to the underlying platform. In Linux key generation is invoked via the openssl CLI command. In Windows key generation is invoke ia the "Certificates - Current User" Windows MMC snap-in as a "Custom Request". In Android, the application invokes the Android Keystore provider for key generation.

### 6.1.4 FCS_CKM.2: Cryptographic Key Establishment

The TOE implements the following key establishment schemes for IPsec:

- Elliptic curve-based key establishment schemes that meets NIST Special Publication 800-56A, “Recommendation for Pair-Wise Key Establishment Schemes Using Discrete Logarithm Cryptography” for Diffie-Hellman groups 19 and 20.
- Key establishment scheme using Diffie-Hellman group 14 that meets RFC 3526, Section 3.

### 6.1.5 FCS_CKM.1: Cryptographic Key Generation Services

The TOE implements asymmetric key generation. The cryptographic functions (including key generation) are implemented by the Aruba Common Cryptographic Module. Table 12: Key Purpose/Key Storage/Key Destruction describes the keys and secrets used by the TOE.

### 6.1.6 FCS_CKM_EXT.2: Cryptographic Key Storage

Table 12 identifies the purpose of each key used by the TOE, the key storage location, and the key destruction method when the key is no longer required. Note: All keys are for IKE/IPsec and are zeroised (overwritten with zeroies) when no longer in used by the IKE/IPsec function.

*Table 12: Key Purpose/Key Storage/Key Destruction*

| Key Name: | Origin/Purpose: | Storage Location: | Key Destruction: |
|---|---|---|---|
| DH Private Components | Used to derive the secret session key during DH key agreement protocol Group 14 (384 bits) ` | Temporarily in volatile RAM | An application program which uses the API may destroy the key. The TOE zeroizes (zero overwrite) this CSP. |
| DRNG Seed Key | DRBGs for key generation | Temporarily in volatile RAM | An application program which uses the API may destroy the key. The |
| | DRBG Seed: SP800-90a DRBG (384 bits)<br><br>DRBG Key: SP800-90a (256 bits) | | TOE zeroizes (zero overwrite) this CSP. |

| Key Name: | Origin/Purpose: | Storage Location: | Key Destruction: |
|---|---|---|---|
| RSA Private Key | Used to create RSA digital signatures key size: 2048 bits | Temporarily in volatile RAM (while in use), persistently in platform key storage (while not in use) | An application program which uses the API may destroy the key. The TOE zeroizes (zero overwrite) this CSP when residing in memory; when persistently stored in platform key storage, it is destroyed by the platform. The TOE has no mechanism to initiate the destruction of persistently stored certificates. |
| ECDSA Private Key | Used to create DSA digital signatures NIST curves: P-256, P- 384 | Temporarily in volatile RAM (while in use), persistently in platform key storage (while not in use) | An application program which uses the API may destroy the key. The TOE zeroizes (zero overwrite) this CSP when residing in memory; when persistently stored in platform key storage, it is destroyed by the platform. The TOE has no mechanism to initiate the destruction of persistently stored certificates. |
| AES Keys | Used during AES encryption, decryption<br><br>Key sizes: 128 bits, 256 bits | Temporarily in volatile RAM | An application program which uses the API may destroy the key. The TOE zeroizes (zero overwrite) this CSP. |
| HMAC Keys | Used during HMAC-SHA-1, HMAC-SHA-256, HMAC-384 operations<br><br>Key sizes: 160 bits, 256 bits, 384 bits | Temporarily in volatile RAM | An application program which uses the API may destroy the key. The TOE zeroizes (zero overwrite) this CSP. |
| DH Public Component | Used to derive the secret session key during DH key agreement protocol DH Groups: Group 14 (2048 bits) | Temporarily in volatile RAM | No longer needed by trusted channel |

| Key Name: | Origin/Purpose: | Storage Location: | Key Destruction: |
|---|---|---|---|
| ECDH Public Componen t | Used to derive the secret session key during ECDH key agreement protocol Group 19 (P-256), Group 20 (P-384) | Temporarily in volatile RAM | N/A |
| RSA Public Keys | Used to verify RSA Signatures key size: 2048 bits | Temporarily in volatile RAM | N/A |
| ECDSA Public Keys | Used to verify ECDSA Signatures NIST Curves: P-256, P-384 | Temporarily in volatile RAM | N/A |

Table 12 lists all the keys manipulated by the TOE. The TOE does not store these keys unencrypted into persistent storage. While the TOE manipulates keys, on Windows and Android, the TOE platform's key storage is used: Windows Certificate Store and Android Keystore, respectively. In Linux, the TOE encrypts the credentials as per FCS_COP.1/SKC. The key storage repositories for each platform are as follows:

### 6.1.7 FCS_CKM_EXT.4: Cryptographic Key Destruction

The TOE and its platform collectively zeroize all plaintext secret and private cryptographic keys and CSPs when no longer required. The cryptographic keys are destroyed when they are no longer in use by the system. Section 6.1.6, Table 12, details how each key is destroyed. The details apply to all platforms.

### 6.1.8 FCS_COP.1/SKC: Cryptographic Operation – Encryption/Decryption

The TOE implements the IPsec protocol ESP as defined by RFC 4303 using the cryptographic algorithms AES-GCM-128, AES-GCM-256 as specified in RFC 4106, and AES-CBC-128, AES-CBC-256 both specified by RFC 3602.

### 6.1.9 FCS_COP.1/Hash: Cryptographic Operation - Hashing

The TOE uses the hash function with other application cryptographic functions. Specifically, SHA-1 and SHA-256 are used as the hash algorithms for the HMAC-SHA-1 and HMAC-SHA-256 message authentication function for ESP. Similarly, SHA-1, SHA-256, and SHA-384 are used as hash algorithms for the corresponding HMAC algorithms used as message authentication for IKE. Users not wishing to rely on SHA-1 for the security of ESP and IKE should refer to the guidance provided in Section 6.1.11.

The TOE also uses SHA-256 and SHA-384 hashing for public key authentication for IKE peer authentication.

Additionally, TOE software updates are verified using a 2048-bit RSA digital signature with SHA-256.

### 6.1.10 FCS_COP.1/Sig: Cryptographic Operation - Signing

The TOE performs cryptographic signature services (generation and verification) in accordance with the following cryptographic algorithms:

- RSA schemes using cryptographic key sizes of 2048-bits that meets FIPS PUB 186-5, "Digital Signature Standard (DSS)", Section 4 is used for IPsec authentication and TOE update verification.
- ECDSA schemes using "NIST curves" P-256 and P-384 that meet FIPS PUB 186-5, "Digital Signature Standard (DSS)", Section 5 is used for IPsec authentication.

### 6.1.11 FCS_COP.1/KeyedHash: Cryptographic Operation - Keyed-Hash Message Authentication

The TOE provides the HMAC-SHA-1, HMAC-SHA-256 and HMAC-SHA-384 algorithms. Refer to Table 11 for the corresponding CAVP certificate demonstrating compliance with these algorithms. These keyed-hash functions can be defined for use in an IPsec connection. The HMAC-SHA-1, HMAC-SHA-256 and HMAC-SHA-384 algorithms are used with key sizes and block sizes of 160, 256 and 384-bits respectively, producing output MAC lengths equal to the block size.

HMAC-SHA-1 can be selected for message authentication in ESP in combination with AES-CBC-128 or AES-CBC-256. Users not wishing to rely on SHA-1 for authentication in ESP, could select HMAC-SHA-256 instead, or choose the authenticated encryption algorithm AES-GCM.

HMAC-SHA-1 can be used for message authentication in IKE in combination with AES-CBC-128 or AES-CBC-256. Users not wishing to rely on SHA-1 for authentication in IKE, should choose instead HMAC-SHA-256, or HMAC-SHA-384.

### 6.1.12 FCS_IPSEC_EXT.1: IPsec

The TOE implements its own cryptography via the Aruba Common Crypto Module (ACCM). The cryptographic functionality is implemented by the application itself as a separate executable that is bundled with the platform. The TOE interacts with the network stack of the platforms using sockets through the respective platform APIs.

The TOE implements the IPsec protocol as specified in RFC 4301; however, the TOE relies upon the VPN Gateway to ensure that the cryptographic algorithms and key sizes negotiated during the IKEv2 negotiation ensure that the security strength of the IKE_SA is greater than or equal to that of the CHILD_SA[16].

The IKEv2 SA lifetimes are configured by the VPN Gateway based upon the length of time. Length of time includes 24 hours or less for Phase 1 and 8 hours or less for Phase 2.

The TOE implements IKEv2, in tunnel mode only. The TOE does not offer transport mode as a configuration option. IKEv2 is implemented as defined in RFCs 7296 (with mandatory support for NAT traversal as specified in section 2.23), RFC 8784, RFC 8247.

The TOE implements peer authentication using RSA certificates or ECDSA certificates that conform to RFC 4945. If certificates are used, the TOE ensures that the distinguished name (DN) contained in a certificate matches the expected DN for the entity attempting to establish a connection and ensures that the certificate has not been revoked (using the Online Certificate Status Protocol [OCSP] in accordance with RFC 6960).

During the Peer Authentication stage of IPsec, the TOE will verify the authenticity of the VPN gateway's X.509v3 certificate by validating the certificate, the certificate path, the certificates revocation status using OCSP, the certificate path terminates in a trusted CA certificate, and that the CA certificate has the basicConstraints extension present and the CA flag set to true.

The SHA hash algorithm (all claimed key sizes) is used as part of HMAC but is also used independently as part of digital signature creation and verification. The TOE generates RSA and ECDSA signatures during IKEv2 peer authentication. The TOE verifies RSA & ECDSA signatures during IKEv2 peer authentication and trusted updates.

[16] Note that the algorithm negotiated will be AES because that is the only available algorithm, so strength is based solely upon key size where more bits are stronger.

The TOE implements various HMAC algorithms (HMAC-SHA1, HMAC-SHA-256) to be used for authentication in ESP in combination with AES-CBC-128 or AES-CBC-256. The specific algorithms used depend upon the ciphersuite being used. The TOE implements the authenticated encryption methods:

- AES-GCM-128
- AES-GCM- 256
- AES-CBC-128 or AES-CBC-256 in combination with HMAC-SHA1 or HMAC-SHA-256 as the authentication algorithm.

As configured on the VPN Gateway, the TOE supports the following Diffie-Hellman (DH) groups for use in SA negotiation:

- 14 (2048-bit MODP),
- 19 (256-bit Random ECP), and
- 20 (384-bit Random ECP).

Only one DH group can be configured for a specific policy so there is no negotiation. Different policies can have different groups. Implementation is the same for both versions of IKE.

The TOE generates the secret value x used in the Diffie-Hellman key exchange ('x' in $g^x$ mod p) using the CAVP tested RBG specified in FCS_RBG_EXT.1 and having possible lengths of 224 for group 14, 256 for group 19, or 384 bits for group 20. When a random number is needed for a nonce, the probability that a specific nonce value will be repeated during the life of a specific IPsec SA is less than 1 in $2^{112}$, $2^{128}$, or $2^{192}$.

The TOE implements minimal SPD rules that are defined implicitly through the configuration and connection of a VPN session. The TOE does not support direct editing of SPD rules. As part of initial configuration the administrator must manually configure the SPD such that if no “rules” are found to match, a final rule exists that causes the network packet to be discarded. The PROTECT and BYPASS rules are implicit and are implemented by configuring a split tunnel. The administrator may configure PROTECT and BYPASS rules by enabling or disabling split-tunnel mode in the VIA connection profile on the VPN Gateway.

All traffic originating from the client operating system is passed through the tunnel established by the VIA client to the VPN Gateway. When split-tunneling is enabled, the VPN Gateway pushes routes configured with the tunnel address command in the VIA connection profile on the VPN Gateway to the VIA client. Traffic matching the routes is forwarded through the IPsec tunnel. The DISCARD rule is not supported by the TOE and must be provided by firewall rules configured on the VPN Gateway and the Platform.

### 6.1.13 FCS_RBG_EXT.1: Random Bit Generation, FCS_RBG_EXT.2: Random Bit Generation from Application

The application implements the DRBG functionality. Therefore FCS_RBG_EXT.2 is included in the security target. The TOE implements SP 800-90A using CTR_DRBG (AES) for all deterministic random bit generation services.

The TOE includes an AES-256 CTR_DRBG (irrespective of the underlying Platform) that seeds itself with 384 bits of entropy (composed of a 256 bit entropy input and a 128 bit nonce) drawn from a platform function intended to provide cryptographically secure randomness (and conditioned using SHA-1). The following list describes the mechanism by which VIA obtains its seeding.

- Android - /dev/random
- Windows - BCryptGenRandom()
- Linux - getrandom() and that it uses a flag of GRND_RANDOM and behaves exactly as /dev/random.

Based on NIAP’s “Clarification to the Entropy Documentation and Assessment Annex”, HPE Aruba Networking assumes a minimum entropy of 0.67 bits of entropy per bit of data from platform provided sources. This minimum- entropy estimate along with knowledge that the seed is 384 bits means that the TOE seeds itself with at least 256 bits of entropy.

### 6.1.14 FCS_STO_EXT.1 Storage of Credentials

The TOE maintains one type of credentials: certificates. For certificates, platform storage mechanisms are used for Windows and Android. Specifically, the Windows version of the TOE uses the Windows Certificate Store while the Android version of the TOE uses the Android keystore. The Linux version of the TOE encrypts directly the certificates, which it stores in its own credential repository.

## 6.2 User Data Protection (FDP)

### 6.2.1 FDP_DAR_EXT.1: Encryption of Sensitive Application Data

The TOE does not implement functionality that allows to store in-non-volatile memory sensitive application data except for the private keys of X509 certificates which are protected by FCS_STO_EXT.1.

### 6.2.2 FDP_DEC_EXT.1: Access to Platform Resources

Sensitive information repositories are defined as those collections of sensitive data that could be expected to be shared among some applications, users, or user roles, but to which not all of these would ordinarily require access.

The TOE restricts its access to using network connectivity when it is needed to communicate to the VPN Gateway.

### 6.2.3 FDP_NET_EXT.1: Network Communications

The TOE allows network communications to be initiated by the user in order to connect to the VPN Gateway. The TOE can also provide always-on functionality for application-initiated network communication. The Windows and Linux TOE platforms provide always-on functionality for application- initiated network communication. However, this functionality must be enabled by an administrator. The Android TOE platform provides always-on functionality available by default.

### 6.2.4 FDP_RIP.2: Full Residual Information Protection

The TOE ensures that no residual information exists in network packets. When the TOE allocates a new buffer for either an incoming or outgoing a network packet, the new packet data will be used to overwrite any previous data in the buffer. If an allocated buffer exceeds the size of the packet, additional space will be overwritten (padded) with zeros before the packet is forwarded (to the external network or delivered to the appropriate, internal application).

The clearing of residual information is processed the same on the Android, Windows, and Linux platforms.

## 6.3 Identification and Authentication

### 6.3.1 FIA_X509_EXT.1: X.509 Certificate Validation

The TOE can use X.509 certificates for authentication. The TOE uses a user specified certificate when attempting to establish the VPN connection. The Android, Linux, and Windows versions of the TOE validate authentication certificates (including the full path) and checks their revocation status using OCSP.

The TOE processes a VPN connection to a server by first comparing the Identification (ID) Payload received from the server against the certificate sent by the server, and if the DN of the certificate does not match the ID, then the TOE does not establish the connection. Assuming the server's certificate matches the ID, the TOE then validates that it can construct a certificate path from the server's certificate through any intermediary CAs to the CA certificate specified by the user in the VPN configuration. If the TOE can successfully build the certificate path, then the TOE will next check the validity of the certificates (e.g., checking its validity dates and that the CA flag is present in the basic constraints section for all CA certs). Assuming the certificates are valid, the TOE finally checks the revocation status of all certificates (starting with the server's certificate and working up the chain). Configuration settings applied by the HPE Aruba Networking Mobility Controller will determine if a certificate is accepted or rejected if the connection to the OCSP server cannot be established; in the evaluated configuration, a configuration will be applied to the TOE that rejects the certificate in these cases.

The TOE implements X.509 certificate validation functionality for the following:

- IPsec – Windows, Android, Linux.

### 6.3.2 FIA_X509_EXT.2: X.509 Certificate Authentication

A certificate is assigned to each VPN profile by an administrator. The VPN profile is downloaded from the server to the client. The VPN Profile tab displays the following information about each downloaded VPN profile:

- Profile: Name of the VPN profile, and the date and time that the profile was added.
- Authentication: IKE protocol version and authentication type.
- Server: IP address of the VPN server.
- Auth Profile: Web authentication profile.
- Certificate: VPN connection certificate (only for certificate-based authentication).

The TOE uses X.509v3 certificates for authentication as defined in RFC 5280 in IPsec exchanges and rejects any certificates that cannot be validated as described above. The certificate will not be accepted when the connection to OCSP server cannot be reached.

## 6.4 Security Management

### 6.4.1 FMT_CFG_EXT.1: Secure by Default Configuration

The TOE is not provided with any default credentials. All external configuration comes from the Mobility Controller. The configuration options for the TOE consists of the URL of the gateway, entering credentials, and the profile prior to establishing a connection.

The TOE requires the following permissions:

- Windows: no end user modification is permitted,

- Linux and Android: all require root permission.

### 6.4.2 FMT_MEC_EXT.1: Supported Configuration Mechanism[17]

The TOE configuration options come from the HPE Aruba Networking Mobility Controller and deploy platform-specific options. The configuration options are stored and set using the mechanisms supported by the following platforms.

- Windows: Registry (HKCU\SOFTWARE\ARUBANETWORKS), C:\ProgramData, and an XML file in %USERPROFILE%.
- Android: /data/data/com.aruba.via/files/ - controller IP address and credential information is stored using SharedPreferences.
- Linux: XML file on disk in user profile.

The configuration options for the TOE consists of entering the URL of the gateway, entering credentials, and downloading profile prior to establishing a connection.

### 6.4.3 FMT_SMF.1: Specification of Management Functions

The TOE does not support any management functionality as described in the [PP_APP_v1.4]. The TOE supports the management functionality as defined in [MOD_VPNC_V2.4].

### 6.4.4 FMT_SMF.1/VPN: Specification of Management Functions (VPN)

The following security management functions are provided by the TOE:

- Specify VPN gateways to use for connections,
- Specify client credentials to be used for connections.

The first time a connection is established, a user opens the VIA client and enters the server name, username, and password. If the VIA web authentication list has more than one VIA authentication profile, the user can choose a VIA authentication profile from the available ones. After successful authentication, the VIA client downloads the appropriate VIA connection profile and establishes the IPsec connection if the user is connected to an untrusted network.

## 6.5 Privacy

### 6.5.1 FPR_ANO_EXT.1: User Consent for Transmission of Personally Identifiable Information

The TOE does not transmit personally identifiable information over a network.

## 6.6 Protection of the TSF

### 6.6.1 FPT_AEX_EXT.1: Anti-Exploitation Capabilities

For each platform, 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 details of each specific platform is identified below.

- The Windows VPN client 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. By default, /DYNAMICBASE is enabled to support ASLR. The TOE does not write user-modifiable

[17] Modified per TD0543 (Windows 10)

files to directories that contain executable files. No files are downloaded to Program Files. The application is built with stack-based buffer overflow protection enabled using the */GS* flag for Windows. The Windows VPN client application is compatible with security features provided by the platform vendor.

- The Android VPN Client 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 – fpic flag is set to support ASLR. The TOE does not write user-modifiable files to directories that contain executable files. No files are downloaded to /data/data/package. The application is built with stack-based buffer overflow protection enabled using –fstack-protector- strong. The Android VPN client application is compatible with security features provided by the platform vendor.

- The Linux VPN Client 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. ASLR is supported by default. By default, 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 using -fpic. The Linux VPN client application is compatible with security features with SELinux.

### 6.6.2 FPT_API_EXT.1: Use of Supported Services and APIs

The TOE includes the use only documented platform APIs. The Platform APIs are documented below.

- **Windows:** See Appendix A.
- **Android:** See Appendix B.
- **Linux:** See Appendix C.

### 6.6.3 FPT_IDV_EXT.1: Software Identification and Versions

HPE Aruba Networking provides a version control system for its software components. The TOE has a unique software versioning that identifies major versions and their subsequent maintenance releases in the following form: <major>.<minor>.<patch>.<build>. Major and minor releases introduce new major and minor features for the product, and patch and build releases identify bug and security patches and the build number.

### 6.6.4 FPT_LIB_EXT.1: Use of Third-Party Libraries

The TOE is packaged with the following third-party libraries.

- Windows: openssl (libeay32.dll, ssleay32.dll), System.Windows.Interactivity, PageAnimator.dll
- Linux: libanmocana.so, opensc-pkcs11.so, pkcs11-engine.so

### 6.6.5 FPT_TST_EXT.1/VPN(1): TSF Self-Test (VPN Client) (Windows and Linux) and FPT_TST_EXT.1/VPN(2): TSF Self-Test (VPN Client) (Android)

The TOE performs known answer power on self-tests (POST) on its cryptographic algorithms to ensure that they are functioning correctly. The TOE utilizes the Aruba Common Cryptographic Module (ACCM) library which implements known answer tests on its cryptographic algorithms to ensure they are working correctly. These known answer tests involve using the ACCM library functions to encrypt blocks of data and comparing the resulting encrypted block of data to a block that is known to be correct. The result of encrypting a block of data is the same every time if the encryption library operates properly. These tests cover the following algorithms, known answer tests, and pairwise consistency tests:

- AES-GCM – 128 bits, 256 bits,
- AES-CBC – 128 bits, 256 bits,

- SHA 1, SHA 256, SHA 384,
- HMAC-SHA1, HMAC-SHA-256, HMAC-SHA-384,
- RSA Pairwise Consistency Test,
- RSA Encrypt/Decrypt Known Answer Test,
- DSA Pairwise Consistency Test,
- ECDSA Pairwise Consistency Test,
- ECDH Pairwise Consistency Test,
- DH Pairwise Consistency Test, and
- DRBG Known Answer Test.

The TOE invokes these self-tests of the ACCM library at start to ensure that those cryptographic algorithms are working correctly.

A self-test is additionally performed on the cryptographic executable code. For Windows and Linux platforms, the cryptographic module is signed by HPE Aruba Networking using 2048-bit RSA. For Android, the entire application is signed by Google as part of its distribution on the Google Play store. This signature uses 4096-bit RSA.

If any self-test fails, the TOE will not start. Successful execution of self-testing is implicit through the application starting without error. For troubleshooting purposes, a debug log can be enabled that writes the outcome of the self-test to a local file that shows the results of the self-test. Since the success or failure of the self-test is implicit in whether or not the application starts successfully, the debug log is only used to confirm the reason behind any anomalous behavior for reporting to HPE Aruba Networking Support.

- Windows:
  - Success: “FIPS Powerup Self Finished Successfully”
  - Failure: “FIPS_powerupSelfTest() failed error <code>”
- Android:
  - Success: “Finished Mocana initialization...”
  - failure: “Mocana initialization error <code>”
- Linux:
  - Success: “Power up self test passed...”
  - Failure: “!!ERROR!! :Power up self test failed: <error code>”

The entire purpose of the TSF is to establish IPsec connectivity. Any tampering of the IPsec functionality, either as a whole or through corruption of individual cryptographic algorithms, would be detected through a failure of the software integrity test or of the individual cryptographic known-answer/pairwise consistency tests. Therefore, the testing provided by the TOE (and its underlying platform in the case of Android) is sufficient to ensure that the application cannot be modified without detection in a way that would compromise the behavior of the TSF.

### 6.6.6 FPT_TUD_EXT.1(1) (Android): Trusted Update, FPT_TUD_EXT.1(2) (Windows and Linux): Trusted Update FPT_TUD_EXT.2: Integrity for Installation and Update

The Abouttab on the TOE shows the current system image version number.

The TOE platforms support loading updates by the administrator. For Android version, the application and signature are provided and verified by the Google Play Store. The Android version is signed through the app store using RSA 2048 bit with SHA-256 digital signature. The TOE platform checks the signature against the downloaded application when the update is obtained by the TOE platform.

For Windows and Linux platforms, the administrator obtains the update in the form of an installer through the HPE Aruba Networking Mobility Controller or the Aruba Support Portal. Regardless of its origin, the update is verified using an RSA 2048 digital signature with SHA-1, signed by HPE Aruba Networking. The installer automatically verifies the digital signature on the update during the installation. An unverified update cannot be installed.

The authorized source for the Windows and Linux updates is from the HPE Networking Support Portal. The Google Play Store is the authorized source for the Android update.

The TOE application package is delivered in the following platform-supported formats:

- Windows: .msi
- Android: .apk, playstore
- Linux: .deb, .rpm

The TOE does not download, modify, replace or update its own binary code. The application is packaged such that its removal results in the deletion of all traces of the application, with the exception of configuration settings and output files.

## 6.7 Trusted Path/Channels

### 6.7.1 FTP_DIT_EXT.1: Protection of Data in Transit

The application encrypts all transmitted sensitive data with IPsec between itself and another trusted IT product. The purpose of the application is to enable a secure IPsec connection between its underlying host OS platform and a remote system that is protected by a VPN gateway. Therefore, any data that traverses the IPsec tunnel is network traffic generated by the underlying OS platform or other applications running on it.

## 6.8 ALC_TSU_EXT.1: Timely Security Updates

HPE Aruba Networking has a Security Incident Response Team (SIRT) that security information can be provided to at https://www.arubanetworks.com/support-services/sirt/. Findings can be submitted over email using a PGP key or by opening a ticket on support site (HTTPS). HPE Aruba Networking does not disclose any identified vulnerability until a patch is available. If vulnerability is disclosed on a side channel (e.g. a published CVE), high or critical findings are addressed as a priority. High vulnerabilities are addressed within 30 days, moderate vulnerabilities within 90 days, and low vulnerabilities within 180 days. Historically, all findings have been addressed within 30 days regardless of severity.

The VIA client can automatically update to newer versions of the VIA client software when the update is made available by the administrator. Note that VIA software for Android devices is managed by the Google application stores and can’t be pushed using the VIA automatic update feature. This feature is enabled by default and can be disabled by unselecting the Allow client to auto-upgrade option in the VIA connection profile.

Windows Clients can be auto upgraded using installation files uploaded on the controller or using an or external downloadable URL. Linux clients can be auto upgraded using external downloadable URL only.

Linux and Windows clients can automatically upgrade their VIA software from a URL defined in the VIA external download URL field in the VIA Connection profile. This option is available for Linux and Windows clients.

# 7 Protection Profile Claims

This TOE is conformant to the following CC specifications:

- *Protection Profile for Application Software, Version 1.4, 2021-10-18* [PP_APP_v1.4] with the optional requirements: FCS_CKM.1/SK, and the selection-based requirements: FCS_CKM.1/AK, FCS_CKM.2, FCS_COP.1(1), FCS_COP.1/Hash, FCS_COP.1/Sig, FCS_COP.1/KeyedHash, FCS_RBG_EXT.2, FIA_X509_EXT.1, FIA_X509_EXT.2, and FPT_TUD_EXT.2.
- *PP-Module for Virtual Private Network (VPN) Clients*, Version 2.4, 2022-03-31 [MOD_VPNC_V2.4].
- *PP-Configuration for Application Software and Virtual Private Network (VPN) Clients*, Version 1.3, 2023-04-07

As explained in Section 3, Security Problem Definition, the Security Problem Definition of the [PP_APP_v1.4] and [MOD_VPNC_V2.4] have been copied verbatim into this ST.

As explained in Section 4, Security Objectives, the Security Objectives of this ST is conformant to [PP_APP_v1.4] and [MOD_VPNC_V2.4] have been copied verbatim into this ST.

The following table identifies all the Security Functional Requirements (SFRs) in this ST. Each SFR is drawn from the [PP_APP_v1.4] and the [MOD_VPNC_V2.4].

*Table 13: SFR Protection Profile Sources*

| Requirement Class | Requirement Component | Source |
|---|---|---|
| **FCS: Cryptographic Support** | FCS_CKM.1/AK: Cryptographic Asymmetric Key Generation | PP_APP_v1.4 |
| | FCS_CKM.1/SK: Cryptographic Symmetric Key Generation | PP_APP_v1.4 |
| | FCS_CKM.1/VPN: Cryptographic Key Generation (IKE) | MOD_VPNC_V2.4 |
| | FCS_CKM.1: Cryptographic Key Generation Services | PP_APP_v1.4 |
| | FCS_CKM.2: Cryptographic Key Establishment | PP_APP_v1.4 |
| | FCS_CKM_EXT.2: Cryptographic Key Storage | MOD_VPNC_V2.4 |
| | FCS_CKM_EXT.4: Cryptographic Key Destruction | MOD_VPNC_V2.4 |
| | FCS_COP.1(1): Cryptographic Operation – Encryption/Decryption | PP_APP_v1.4 |
| | FCS_COP.1/Sig: Cryptographic Operation - Signing | PP_APP_v1.4 |
| | FCS_COP.1/Hash: Cryptographic Operation - Hashing | PP_APP_v1.4 |
| | FCS_COP.1/KeyedHash: Cryptographic Operation - Keyed-Hash Message Authentication | PP_APP_v1.4 |
| | FCS_COP.1/SKC: Cryptographic Operation – Encryption/Decryption | PP_APP_v1.4 |
| | FCS_IPSEC_EXT.1: IPsec | MOD_VPNC_V2.4 |

| Requirement Class | Requirement Component | Source |
|---|---|---|
| | FCS_RBG_EXT.1: Random Bit Generation Services | PP_APP_v1.4 |
| | FCS_RBG_EXT.2: Random Bit Generation from Application | PP_APP_v1.4 |
| | FCS_STO_EXT.1: Storage of Credentials (iterated by ST author) | PP_APP_v1.4 |
| **FDP: User Data Protection** | FDP_DEC_EXT.1: Access to Platform Resources | PP_APP_v1.4 |
| | FDP_NET_EXT.1: Network Communications | PP_APP_v1.4 |
| | FDP_DAR_EXT.1: Encryption Of Sensitive Application Data | PP_APP_v1.4 |
| | FDP_RIP.2: Full Residual Information Protection | MOD_VPNC_V2.4 |
| **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 |
| | FMT_SMF.1/VPN: Specification of Management Functions (VPN) | MOD_VPNC_V2.4 |
| **FPR: Privacy** | FPR_ANO_EXT.1: User Consent for Transmission of Personally Identifiable Information | PP_APP_v1.4 |
| **FPT: Protection of the TSF** | FPT_API_EXT.1: Use of Supported Services and APIs | PP_APP_v1.4 |
| | FPT_AEX_EXT.1: Anti-Exploitation Capabilities | 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: Trusted Update (iterated by ST author) | PP_APP_v1.4 |
| | FPT_TUD_EXT.2: Integrity for Installation and Update | PP_APP_v1.4 |
| | FPT_TST_EXT.1/VPN: TSF Self-Test (VPN Client) (iterated by ST author) | MOD_VPNC_V2.4 |
| **FTP: Trusted Path/Channels** | FTP_DIT_EXT.1: Protection of Data in Transit | MOD_VPNC_V2.4 |

# 8 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 set of security functions work together to satisfy all of the security functions and assurance 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, the 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 the security requirements. The security functions described in the TOE summary specification are all necessary for the required security functionality in the TSF. Table 14: Security Functions vs. Requirements Mapping demonstrates the relationship between security requirements and security functions.

*Table 14: 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.1 | X | | | | | | |
| FCS_CKM_EXT.2 | X | | | | | | |
| FCS_CKM.1/AK | X | | | | | | |
| FCS_CKM.1/SK | X | | | | | | |
| FCS_CKM.1/VPN | X | | | | | | |
| FCS_CKM.2 | X | | | | | | |
| FCS_CKM_EXT.4 | X | | | | | | |
| FCS_COP.1(1) | X | | | | | | |
| FCS_COP.1/Hash | X | | | | | | |
| FCS_COP.1/Sig | X | | | | | | |
| FCS_COP.1/Keyed Hash | X | | | | | | |
| FCS_IPSEC_EXT.1 | X | | | | | | |
| FCS_STO_EXT.1 | X | | | | | | |
| FCS_RBG_EXT.1 | X | | | | | | |
| FCS_RBG_EXT.2 | X | | | | | | |
| FDP_DEC_EXT.1 | | X | | | | | |
| FDP_DAR_EXT.1 | | X | | | | | |
| FDP_RIP.2 | | X | | | | | |

| | Cryptographic support | User Data Protection | Identification and authentication | Security management | Privacy | Protection of the TSF | Trusted path/channels |
|---|---|---|---|---|---|---|---|
| FIA_X509_EXT.1 | | | X | | | | |
| FIA_X509_EXT.2 | | | X | | | | |
| FMT_CFG_EXT.1 | | | | X | | | |
| FMT_MEC_EXT.1 | | | | X | | | |
| FMT_SMF.1 | | | | X | | | |
| FMT_SMF.1/VPN | | | | X | | | |
| FPR_ANO_EXT.1 | | | | | X | | |
| FPT_API_EXT.1 | | | | | | X | |
| FPT_AEX_EXT.1 | | | | | | X | |
| FPT_IDV_EXT.1 | | | | | | X | |
| FPT_LIB_EXT.1 | | | | | | X | |
| FPT_TUD_EXT.1(1) | | | | | | X | |
| FPT_TUD_EXT.1(2) | | | | | | X | |
| FPT_TUD_EXT.2 | | | | | | X | |
| FPT_TST_EXT.1/VPN | | | | | | X | |
| FTP_DIT_EXT.1 | | | | | | | X |

# Appendix A: Windows Platform APIs

ntdll.dll

KERNEL32.DLL

KERNELBASE.dll

WS2_32.dll

ucrtbase.dll

RPCRT4.dll

USER32.dll

win32u.dll

GDI32.dll

gdi32full.dll

msvcp_win.dll

ADVAPI32.dll

msvcrt.dll

sechost.dll

SHELL32.dll

wintypes.dll

combase.dll

ole32.dll

OLEAUT32.dll

CRYPT32.dll

bcrypt.dll

WTSAPI32.dll

MSVCP140.dll

VCRUNTIME140.dll

VCRUNTIME140_1.dll

dbghelp.dll

IPHLPAPI.DLL

VERSION.dll

SspiCli.dll

WININET.dll

dbgcore.DLL

CRYPTSP.dll

rsaenh.dll

CRYPTBASE.dll

bcryptPrimitives.dll

windows.storage.dll

SHCORE.dll

shlwapi.dll

profapi.dll

ntmarta.dll

WINSTA.dll

DPAPI.dll

kernel.appcore.dll

clbcatq.dll

mswsock.dll

NSI.dll

Wintrust.DLL

MSASN1.dll

imagehlp.dll

gpapi.dll

cryptnet.dll

WINNSI.DLL

ncrypt.dll

NTASN1.dll

DNSAPI.dll

fwpuclnt.dll

rasadhlp.dll

WINHTTP.dll

dhcpcsvc6.DLL

dhcpcsvc.DLL

webio.dll

Cabinet.dll

# Appendix B: Android Platform APIs

android.animation.Animator

android.annotation.SuppressLint

android.annotation.SuppressLint

android.annotation.TargetApi

android.app.Activity

android.app.Activity

android.app.AlertDialog

android.app.Application

android.app.Application

android.app.Dialog

android.app.KeyguardManager

android.app.KeyguardManager

android.app.ListActivity

android.app.ListActivity

android.app.PendingIntent

android.app.PendingIntent

android.app.ProgressDialog

android.app.Service

android.app.Service

android.content.ActivityNotFoundException

android.content.BroadcastReceiver

android.content.ComponentName

android.content.ComponentName

android.content.ContentResolver

android.content.Context

android.content.Context

android.content.DialogInterface.OnCancelListener;

android.content.DialogInterface.OnClickListener

android.content.DialogInterface

android.content.DialogInterface

android.content.Intent

android.content.Intent

android.content.IntentFilter

android.content.IntentFilter

android.content.pm.PackageInfo

android.content.pm.PackageInfo

android.content.pm.PackageManager.NameNotFoundException

android.content.pm.PackageManager.NameNotFoundException

android.content.pm.PackageManager

android.content.pm.PackageManager

android.content.res.Configuration

android.content.res.Resources

android.content.RestrictionsManager

android.content.ServiceConnection

android.content.ServiceConnection

android.content.SharedPreferences.Editor

android.content.SharedPreferences

android.database.Cursor

android.graphics.Bitmap

android.graphics.Bitmap

android.graphics.BitmapFactory

android.graphics.BitmapFactory

android.graphics.Canvas

android.graphics.Color

android.graphics.ColorFilter

android.graphics.drawable.Drawable

android.graphics.drawable.LayerDrawable

android.graphics.Paint

android.graphics.Path

android.graphics.PixelFormat

android.graphics.Rect

android.graphics.RectF

android.graphics.Typeface

android.net.ConnectivityManager

android.net.ConnectivityManager

android.net.http.SslError

android.net.NetworkInfo

android.net.NetworkInfo

android.net.Uri

android.net.VpnService

android.net.VpnService

android.net.wifi.ScanResult

android.net.wifi.SupplicantState

android.net.wifi.WifiConfiguration.KeyMgmt

android.net.wifi.WifiConfiguration

android.net.wifi.WifiInfo

android.net.wifi.WifiManager

android.net.wifi.WifiManager

android.os.AsyncTask

android.os.AsyncTask

android.os.Binder

android.os.Binder

android.os.Build

android.os.Build

android.os.Bundle

android.os.Bundle

android.os.Environment

android.os.Environment

android.os.Handler

android.os.Handler

android.os.IBinder

android.os.IBinder

android.os.Looper

android.os.Looper

android.os.Message

android.os.Message

android.os.Messenger

android.os.Messenger

android.os.ParcelFileDescriptor

android.os.ParcelFileDescriptor

android.os.PersistableBundle

android.os.RemoteException

android.os.RemoteException

android.os.SystemClock

android.os.SystemClock

android.provider.MediaStore

android.provider.Settings.Secure

android.security.KeyChain

android.security.KeyChain

android.security.KeyChainAliasCallback

android.security.KeyChainAliasCallback

android.security.KeyChainException

android.security.KeyChainException

android.security.keystore.KeyInfo

android.support.annotation.AnimRes

android.support.annotation.AnimRes

android.support.annotation.LayoutRes

android.support.annotation.Nullable

android.support.annotation.RequiresApi

android.support.annotation.RequiresApi

android.support.annotation.StringRes

android.support.annotation.StringRes

android.support.design.widget.TabLayout

android.support.v4.app.ActivityCompat

android.support.v4.app.Fragment

android.support.v4.app.FragmentManager

android.support.v4.app.FragmentPagerAdapter

android.support.v4.app.FragmentTransaction

android.support.v4.content.ContextCompat

android.support.v4.content.FileProvider

android.support.v4.view.ViewPager

android.support.v7.app.AppCompatActivity

android.support.v7.widget.Toolbar

android.system.ErrnoException

android.system.Os

android.text.Editable

android.text.format.Formatter

android.text.format.Formatter

android.text.InputType

android.text.method.LinkMovementMethod

android.text.TextUtils

a n d r oid.text.TextUtils

android.text.TextWatcher

android.util.AttributeSet

android.util.DisplayMetrics

android.util.Log

android.util.Log

android.view.animation.Animation

android.view.animation.Animation

android.view.animation.AnimationUtils

android.view.animation.AnimationUtils

android.view.animation.RotateAnimation

android.view.inputmethod.InputMethodManager

android.view.inputmethod.InputMethodManager

android.view.KeyEvent

android.view.LayoutInflater

android.view.Menu

android.view.MotionEvent

android.view.View.OnClickListener

android.view.View.OnClickListener

android.view.View

android.view.View

android.view.ViewGroup

android.view.ViewGroup

android.webkit.CookieManager

android.webkit.SslErrorHandler

android.webkit.WebResourceError

android.webkit.WebResourceRequest

android.webkit.WebResourceResponse

android.webkit.WebView

android.webkit.WebViewClient

android.widget.AdapterView.OnItemClickListener android.widget.AdapterView

android.widget.ArrayAdapter

android.widget.BaseAdapter

android.widget.Button

android.widget.Button

android.widget.CheckBox

android.widget.CheckBox

android.widget.CheckedTextView

android.widget.CompoundButton.OnCheckedChangeListener

android.widget.CompoundButton.OnCheckedChangeListener

android.widget.CompoundButton

android.widget.CompoundButton

android.widget.EditText

android.widget.EditText

android.widget.FrameLayout

android.widget.ImageView

android.widget.LinearLayout

android.widget.ListView

android.widget.ListView

android.widget.RadioButton

android.widget.RadioGroup

android.widget.RelativeLayout

android.widget.ScrollView

android.widget.ScrollView

android.widget.Switch

android.widget.Switch

android.widget.TextView

android.widget.TextView

android.widget.Toast

android.widget.Toast

android.widget.ViewSwitcher

com.sec.enterprise.mdm.services.vpn.knoxvpn.IKnoxVpnService

com.sec.vpn.knox.GenericVpnContext

com.sec.vpn.knox.GenericVpnContext

java.awt.datatransfer.DataFlavor

java.io.*

java.io.BufferedInputStream

java.io.BufferedInputStream

java.io.BufferedOutputStream

java.io.BufferedOutputStream

java.io.BufferedReader

java.io.BufferedWriter

java.io.ByteArrayInputStream

java.io.ByteArrayInputStream

java.io.ByteArrayOutputStream

java.io.ByteArrayOutputStream

java.io.DataInputStream

java.io.DataOutputStream

java.io.DataOutputStream

java.io.EOFException

java.io.File

java.io.File

java.io.FileFilter

java.io.FileInputStream

java.io.FileInputStream

java.io.FileNotFoundException

java.io.FileNotFoundException

java.io.FileOutputStream

java.io.FileOutputStream

java.io.FilterInputStream

java.io.FilterOutputStream

java.io.InputStream

java.io.InputStream

java.io.InputStreamReader

java.io.InputStreamReader

java.io.IOException

java.io.IOException

java.io.LineNumberReader

java.io.ObjectInputStream

java.io.ObjectInputStream

java.io.ObjectOutput

java.io.ObjectOutputStream

java.io.ObjectOutputStream

java.io.OutputStream

java.io.OutputStream

java.io.OutputStreamWriter

java.io.PipedInputStream

java.io.PipedOutputStream

java.io.PrintStream

java.io.PushbackInputStream

java.io.Reader

java.io.Serializable

java.io.StreamCorruptedException

java.io.UnsupportedEncodingException

java.io.Writer

java.lang.reflect.Constructor

java.math.*

java.math.BigInteger

java.math.BigInteger

java.net.ConnectException

java.net.CookieStore

java.net.CookieStore

java.net.HttpCookie

java.net.HttpCookie

java.net.HttpURLConnection

java.net.Inet4Address

java.net.InetAddress

java.net.InetAddress

java.net.InetSocketAddress

java.net.InetSocketAddress

java.net.NetworkInterface

java.net.NetworkInterface

java.net.ServerSocket

java.net.Socket

java.net.Socket

java.net.SocketAddress

java.net.SocketAddress

java.net.SocketException

java.net.SocketException

java.net.UnknownHostException

java.net.UnknownHostException

java.net.URI

java.net.URISyntaxException

java.net.URL

java.net.URL

java.net.URLClassLoader

java.nio.BufferUnderflowException

java.nio.ByteBuffer

java.nio.ByteOrder

java.nio.charset.Charset

java.nio.file.FileAlreadyExistsException

java.nio.file.Files

java.nio.file.Paths

java.security.*

java.security.AccessController

java.security.AlgorithmParameterGenerator

java.security.AlgorithmParameterGeneratorSpi

java.security.AlgorithmParameters

java.security.AlgorithmParametersSpi

java.security.BasicPermission

java.security.cert.*

java.security.cert.Certificate

java.security.cert.Certificate

java.security.cert.CertificateEncodingException

java.security.cert.CertificateEncodingException

java.security.cert.CertificateException

java.security.cert.CertificateException

java.security.cert.CertificateExpiredException

java.security.cert.CertificateExpiredException

java.security.cert.CertificateFactory

java.security.cert.CertificateFactory

java.security.cert.CertificateFactorySpi

java.security.cert.CertificateNotYetValidException

java.security.cert.CertificateNotYetValidException

java.security.cert.CertificateParsingException

java.security.cert.CertPath

java.security.cert.CertPathBuilder

java.security.cert.CertPathBuilderException

java.security.cert.CertPathBuilderResult

java.security.cert.CertPathBuilderSpi

java.security.cert.CertPathParameters

java.security.cert.CertPathValidator

java.security.cert.CertPathValidatorException

java.security.cert.CertPathValidatorResult

java.security.cert.CertPathValidatorSpi

java.security.cert.CertSelector

java.security.cert.CertStore

java.security.cert.CertStoreException

java.security.cert.CertStoreParameters

java.security.cert.CertStoreSpi

java.security.cert.CollectionCertStoreParameters

java.security.cert.CRL

java.security.cert.CRLException

java.security.cert.CRLSelector

java.security.cert.LDAPCertStoreParameters

java.security.cert.PKIXBuilderParameters

java.security.cert.PKIXCertPathBuilderResult

java.security.cert.PKIXCertPathChecker

java.security.cert.PKIXCertPathValidatorResult

java.security.cert.PKIXParameters

java.security.cert.PolicyNode

java.security.cert.PolicyQualifierInfo

java.security.cert.TrustAnchor

java.security.cert.X509Certificate

java.security.cert.X509Certificate

java.security.cert.X509CertSelector

java.security.cert.X509CRL

java.security.cert.X509CRLEntry

java.security.cert.X509CRLSelector

java.security.cert.X509Extension

java.security.DigestInputStream

java.security.DigestOutputStream

java.security.GeneralSecurityException

java.security.GeneralSecurityException

java.security.interfaces.DSAKey

java.security.interfaces.DSAParams

java.security.interfaces.DSAPrivateKey

java.security.interfaces.DSAPublicKey

java.security.interfaces.ECKey

java.security.interfaces.ECPrivateKey

java.security.interfaces.ECPublicKey

java.security.interfaces.ECPublicKey

java.security.interfaces.RSAPrivateCrtKey

java.security.interfaces.RSAPrivateKey

java.security.interfaces.RSAPublicKey

java.security.interfaces.RSAPublicKey

java.security.InvalidAlgorithmParameterException

java.security.InvalidKeyException

java.security.InvalidKeyException

java.security.InvalidParameterException

java.security.InvalidParameterException

java.security.Key

java.security.Key

java.security.KeyFactory

java.security.KeyFactory

java.security.KeyFactorySpi

java.security.KeyManagementException

java.security.KeyManagementException

java.security.KeyPair

java.security.KeyPairGenerator

java.security.KeyStore.LoadStoreParameter

java.security.KeyStore.ProtectionParameter

java.security.KeyStore

java.security.KeyStore

java.security.KeyStoreException

java.security.KeyStoreException

java.security.KeyStoreSpi

java.security.MessageDigest

java.security.NoSuchAlgorithmException

java.security.NoSuchAlgorithmException

java.security.NoSuchProviderException

java.security.NoSuchProviderException

java.security.Permission

java.security.Principal

java.security.Principal

java.security.PrivateKey

java.security.PrivateKey

java.security.PrivilegedAction

java.security.Provider

java.security.ProviderException

java.security.PublicKey

java.security.PublicKey

java.security.SecureRandom

java.security.SecureRandom

java.security.Security

java.security.Security

java.security.Signature

java.security.Signature

java.security.SignatureException

java.security.SignatureException

java.security.SignatureSpi

java.security.spec.AlgorithmParameterSpec

java.security.spec.DSAParameterSpec

java.security.spec.DSAPrivateKeySpec

java.security.spec.DSAPublicKeySpec

java.security.spec.ECField

java.security.spec.ECFieldF2m

java.security.spec.ECFieldFp

java.security.spec.ECGenParameterSpec

java.security.spec.ECParameterSpec

java.security.spec.ECPoint

java.security.spec.ECPrivateKeySpec

java.security.spec.ECPublicKeySpec

java.security.spec.EllipticCurve

java.security.spec.EllipticCurve

java.security.spec.InvalidKeySpecException

java.security.spec.InvalidKeySpecException

java.security.spec.InvalidParameterSpecException

java.security.spec.KeySpec

java.security.spec.MGF1ParameterSpec

java.security.spec.PKCS8EncodedKeySpec

java.security.spec.PSSParameterSpec

java.security.spec.RSAKeyGenParameterSpec

java.security.spec.RSAPrivateCrtKeySpec

java.security.spec.RSAPrivateKeySpec

java.security.spec.RSAPublicKeySpec

java.security.spec.X509EncodedKeySpec

java.security.UnrecoverableKeyException

java.security.UnrecoverableKeyException

java.sql.Date

java.text.DateFormat

java.text.DateFormat

java.text.DecimalFormat

java.text.Format

java.text.MessageFormat

java.text.ParseException

java.text.ParseException

java.text.SimpleDateFormat

java.text.SimpleDateFormat

java.util.*

java.util.ArrayList

java.util.ArrayList

java.util.Arrays

java.util.Arrays

java.util.Calendar

java.util.Calendar

java.util.Collection

java.util.Collections

java.util.Collections

java.util.concurrent.atomic.AtomicBoolean

java.util.concurrent.ConcurrentHashMap

java.util.concurrent.Executors

java.util.concurrent.Executors

java.util.concurrent.ScheduledExecutorService

java.util.concurrent.ScheduledExecutorService

java.util.concurrent.TimeUnit

java.util.concurrent.TimeUnit

java.util.Date

java.util.Date

java.util.Enumeration

java.util.Enumeration

java.util.HashMap

java.util.HashMap

java.util.HashSet

java.util.HashSet

java.util.Hashtable

java.util.Iterator

java.util.Iterator

java.util.LinkedHashSet

java.util.LinkedList

java.util.List

java.util.List

java.util.ListIterator

java.util.ListIterator

java.util.Locale

java.util.Map

java.util.Map

java.util.MissingResourceException

java.util.Properties

java.util.Random

java.util.regex.Matcher

java.util.regex.Matcher

java.util.regex.Pattern

java.util.regex.Pattern

java.util.ResourceBundle

java.util.Scanner

java.util.Scanner

java.util.Set

java.util.Set

java.util.SimpleTimeZone

java.util.StringTokenizer

java.util.StringTokenizer

java.util.Timer

java.util.Timer

java.util.TimerTask

java.util.TimerTask

java.util.TimeZone

java.util.TimeZone

java.util.Vector

java.util.Vector

java.util.zip.Deflater

java.util.zip.DeflaterOutputStream

java.util.zip.Inflater

java.util.zip.InflaterInputStream

java.util.zip.ZipEntry

java.util.zip.ZipOutputStream

javax.activation.ActivationDataFlavor

javax.activation.CommandMap

javax.activation.DataContentHandler

javax.activation.DataHandler

javax.activation.DataSource

javax.activation.FileDataSource

javax.activation.MailcapCommandMap

javax.crypto.BadPaddingException

javax.crypto.Cipher

javax.crypto.Cipher

javax.crypto.CipherInputStream

javax.crypto.CipherOutputStream

javax.crypto.CipherSpi

javax.crypto.IllegalBlockSizeException

javax.crypto.interfaces.DHKey

javax.crypto.interfaces.DHPrivateKey

javax.crypto.interfaces.DHPublicKey

javax.crypto.interfaces.PBEKey

javax.crypto.KeyAgreement

javax.crypto.KeyAgreementSpi

javax.crypto.KeyGenerator

javax.crypto.KeyGenerator

javax.crypto.KeyGeneratorSpi

javax.crypto.Mac

javax.crypto.MacSpi

javax.crypto.NoSuchPaddingException

javax.crypto.SecretKey

javax.crypto.SecretKeyFactory

javax.crypto.SecretKeyFactory

javax.crypto.SecretKeyFactorySpi

javax.crypto.ShortBufferException

javax.crypto.spec.DESedeKeySpec

javax.crypto.spec.DESedeKeySpec

javax.crypto.spec.DESKeySpec

javax.crypto.spec.DHGenParameterSpec

javax.crypto.spec.DHParameterSpec

javax.crypto.spec.DHPrivateKeySpec

javax.crypto.spec.DHPublicKeySpec

javax.crypto.spec.IvParameterSpec

javax.crypto.spec.IvParameterSpec

javax.crypto.spec.OAEPParameterSpec

javax.crypto.spec.PBEKeySpec

javax.crypto.spec.PBEParameterSpec

javax.crypto.spec.PSource

javax.crypto.spec.RC2ParameterSpec

javax.crypto.spec.RC5ParameterSpec

javax.crypto.spec.SecretKeySpec

javax.crypto.spec.SecretKeySpec

javax.mail.Address

javax.mail.BodyPart

javax.mail.Header

javax.mail.internet.ContentType

javax.mail.internet.InternetAddress

javax.mail.internet.InternetHeaders

javax.mail.internet.MimeBodyPart

javax.mail.internet.MimeMessage

javax.mail.internet.MimeMultipart

javax.mail.internet.MimePart

javax.mail.internet.SharedInputStream

javax.mail.Message

javax.mail.MessagingException

javax.mail.Multipart

javax.mail.Part

javax.mail.Session

javax.mail.Transport

javax.naming.Context

javax.naming.directory.Attribute

javax.naming.directory.DirContext

javax.naming.directory.InitialDirContext

javax.naming.directory.SearchControls

javax.naming.directory.SearchResult

javax.naming.NamingEnumeration

javax.naming.NamingException

javax.net.ssl.HostnameVerifier

javax.net.ssl.HostnameVerifier

javax.net.ssl.HttpsURLConnection

javax.net.ssl.HttpsURLConnection

javax.net.ssl.KeyManager

javax.net.ssl.KeyManager

javax.net.ssl.SSLContext

javax.net.ssl.SSLContext

javax.net.ssl.SSLException

javax.net.ssl.SSLException

javax.net.ssl.SSLSession

javax.net.ssl.SSLSession

javax.net.ssl.SSLSocket

javax.net.ssl.TrustManager

javax.net.ssl.TrustManager

javax.net.ssl.TrustManagerFactory

javax.net.ssl.TrustManagerFactory

javax.net.ssl.X509ExtendedKeyManager

javax.net.ssl.X509ExtendedKeyManager

javax.net.ssl.X509TrustManager

javax.net.ssl.X509TrustManager

javax.security.auth.x500.X500Principal

javax.security.auth.x500.X500Principal

javax.xml.parsers.DocumentBuilder

javax.xml.parsers.DocumentBuilder

javax.xml.parsers.DocumentBuilderFactory

javax.xml.parsers.DocumentBuilderFactory

org.apache.commons.io.IOUtils

org.apache.http.client.CookieStore

org.apache.http.client.entity.UrlEncodedFormEntity

org.apache.http.client.entity.UrlEncodedFormEntity

org.apache.http.client.HttpClient

org.apache.http.client.HttpClient

org.apache.http.client.methods.HttpGet

org.apache.http.client.methods.HttpGet

org.apache.http.client.methods.HttpPost

org.apache.http.client.methods.HttpPost

org.apache.http.client.protocol.ClientContext

org.apache.http.client.protocol.ClientContext

org.apache.http.client.utils.URIUtils

org.apache.http.client.utils.URIUtils

org.apache.http.conn.scheme.PlainSocketFactory

org.apache.http.conn.scheme.PlainSocketFactory

org.apache.http.conn.scheme.Scheme

org.apache.http.conn.scheme.Scheme

org.apache.http.conn.scheme.SchemeRegistry

org.apache.http.conn.scheme.SchemeRegistry

org.apache.http.conn.ssl.SSLSocketFactory

org.apache.http.conn.ssl.SSLSocketFactory

org.apache.http.cookie.Cookie

org.apache.http.cookie.Cookie

org.apache.http.Header

org.apache.http.HttpEntity

org.apache.http.HttpEntity

org.apache.http.HttpResponse

org.apache.http.HttpResponse

org.apache.http.HttpStatus

org.apache.http.HttpStatus

org.apache.http.impl.client.DefaultHttpClient

org.apache.http.impl.client.DefaultHttpClient

org.apache.http.impl.conn.tsccm.ThreadSafeClientConnManager

org.apache.http.impl.conn.tsccm.ThreadSafeClientConnManager

org.apache.http.impl.cookie.BasicClientCookie

org.apache.http.message.BasicNameValuePair

org.apache.http.message.BasicNameValuePair

org.apache.http.NameValuePair

org.apache.http.NameValuePair

org.apache.http.params.BasicHttpParams

org.apache.http.params.BasicHttpParams

org.apache.http.params.HttpConnectionParams

org.apache.http.params.HttpConnectionParams org.apache.http.params.HttpParams

org.apache.http.params.HttpParams

org.apache.http.protocol.BasicHttpContext

org.apache.http.protocol.BasicHttpContext

org.apache.http.protocol.HttpContext

org.apache.http.protocol.HttpContext

org.json.JSONException

org.json.JSONException

org.json.JSONObject

org.json.JSONObject

org.w3c.dom.Document

org.w3c.dom.Document

org.w3c.dom.Element

org.w3c.dom.Element

org.w3c.dom.Node

org.w3c.dom.Node

org.w3c.dom.NodeList

org.w3c.dom.NodeList

# Appendix C: Linux Platform APIs

libdl.so

libxml2.so

libproxy.so

libdbus-1.so

libQt5Widgets.so

libQt5Gui.so

libQt5Network.so

libQt5Core.so

libpthread.so

libstdc++.so

libgcc_s.so

libc.so

ld-linux-x86-64.so

librt.so

libz.so

libicuuc.so

liblzma.so

libm.so

libsystemd.so

libgobject-2.0.so

libglib-2.0.so

libX11.so

libpng12.so

libharfbuzz.so

libGL.so

libicui18n.so

libpcre16.so

libicudata.so

libselinux.so

libgcrypt.so

libffi.so

libpcre.so

libxcb.so

libfreetype.so

libgraphite2.so

libexpat.so

libxcb-dri3.so

libxcb-present.so

libxcb-sync.so

libxshmfence.so

libglapi.so

libXext.so

libXdamage.so

libXfixes.so

libX11-xcb.so

libxcb-glx.so

libxcb-dri2.so

libXxf86vm.so

libdrm.so

libgpg-error.so

libXau.so

libXdmcp.so