SecuSUITE Client (ASPP13/PKGTLS11/VVoIPASEP10) Security Target

Version 0.7
January 30, 2020

Prepared for:
BlackBerry Limited

Prepared By:

Gossamer Laboratories
www.gossamersec.com

1. Security Target Introduction

This section identifies the Security Target (ST) and Target of Evaluation (TOE) identification, ST conventions, ST conformance claims, and the ST organization. The TOE is SecuSUITE Client provided by BlackBerry Limited. The TOE is being evaluated as a Voice/Video over IP (VVoIP) endpoint.

The Security Target contains the following additional sections:

- Conformance Claims (Section 2)
- Security Objectives (Section 3)
- Extended Components Definition (Section 4)
- Security Requirements (Section 5)
- TOE Summary Specification (Section 6)

Conventions

The following conventions have been applied in this document:

- Security Functional Requirements – Part 2 of the CC defines the approved set of operations that may be applied to functional requirements: iteration, assignment, selection, and refinement.
    - Iteration: allows a component to be used more than once with varying operations. In the ST, iteration is indicated by a parenthetical number placed at the end of the component. For example FDP_ACC.1(1) and FDP_ACC.1(2) indicate that the ST includes two iterations of the FDP_ACC.1 requirement.
    - 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 …").
- Other sections of the ST – Other sections of the ST use bolding to highlight text of special interest, such as captions.

1.1 Security Target Reference

ST Title – BlackBerry SecuSUITE v4.0 (ASPP13/PKGTLS11/VVoIPASEP10) Security Target

ST Version – Version 0.7

ST Date – January 30, 2020

1.2 TOE Reference

TOE Identification – SecuSUITE Client v4.0

TOE Developer – BlackBerry Limited

Evaluation Sponsor – BlackBerry Limited

1.3 TOE Overview

The Target of Evaluation (TOE) is SecuSUITE Client v4.0.

The TOE, herein referred to as the SecuSUITE Client or the TOE, is a VoIP application that executes on an Android or iOS mobile device operating system. The TOE executes on the following mobile devices:

a) Samsung Galaxy S9, S9+, S10, S10+, Note9, Note10 (Android 8.0/8.1)

b) Apple iPhone 8, 8 Plus, X, Xs, Xs Max, XR (iOS 12)

1.4 TOE Description

The TOE, herein referred to as the SecuSUITE Client or the TOE, is a VoIP application that executes on an evaluated mobile device operating system

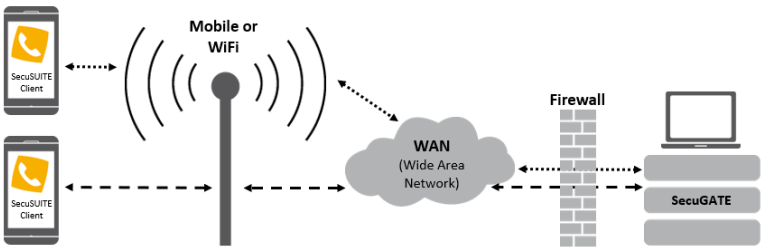

Figure 1-1 TOE Usage

User Context

The TOE user downloads the SecuSUITE Client from an app store (e.g. Apple Store, Google Play) or it is pushed via a Mobile Device Management (MDM) server (e.g. BlackBerry Enterprise Server) and installs the app to their mobile device. On first use of the app, the user must go through a registration process in order to register to a specified BlackBerry SecuGATE (identified by URI).

Once registered, the user can place secure VoIP calls using the app with largely the same interactions as with a normal phone call. The SecuSUITE Client provides encryption of user call signaling and voice data.

Users are typically invited to join SecuSUITE via an activation email initiated by their corporate IT administrator who adds users via the BlackBerry SecuGATE administration portal.

SecuSUITE Context

The TOE is part of the SecuSUITE Security Solution shown in Figure 1-2. The TOE does not work in isolation but relies on BlackBerry SecuGATE components to enable a secure VoIP communication.

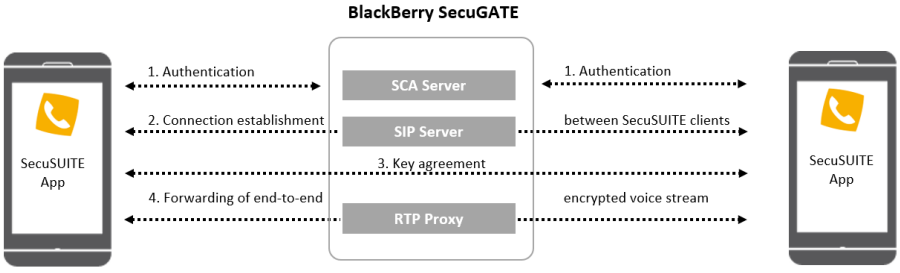

Figure 1-2 SecuSUITE Security Solution

As shown in Figure 1-2, the SecuSUITE VoIP process flow is as follows:

a) Step 1 Initial Registration. Every participating client has to register first to the Secure Client Authentication (SCA) server. The SCA server authenticates users and enrolls required client and user certificates as well as client configuration. Only clients that have been enrolled via the SCA service are able to connect to the SIP server and are allowed to establish end-to-end encrypted communication to other SecuSUITE clients. Note: Clients must also register to the SIP server using a SIP password. This is in addition to initial client registration with the SCA server.

b) Step 2 Connection establishment. The Session Initiation Protocol (SIP) together with TLS is used to establish a secure connection between mobile devices. The use of a TLS connection, providing encryption and mutual authentication, ensures that the devices connect with authorized SIP servers and the dialed call numbers are transmitted encrypted. The BlackBerry SecuGATE SIP Server Security Target defines the SIP Server TOE.

c) Step 3 Key agreement. When a call is placed and accepted, SecuSUITE clients exchange SIP messages that include digital certificates used to confirm caller identity and perform key agreement for SRTP encryption.

d) Step 4 End-to-end encrypted voice communication established. Clients utilize the SRTP protocol to exchange encrypted voice communications. The voice stream remains encrypted while traversing the BlackBerry SecuGATE and only the clients have access to the SRTP session keys.

e) Step 5 Forwarding of end-to-end encrypted voice stream. During connection signaling, the SIP server sets up the RTP/RTCP packet bridging in the Real-Time Transport Protocol (RTP) Proxy for this connection. The RTP Proxy relays / bridges the encrypted data stream between clients. The main purpose of the RTP Proxy is to make the communication between SIP user agents behind NAT/NAPT possible.

VoIP Client

The SecuSUITE Client establishes a secure tunnel for voice communications with another SecuSUITE client or the SecuGATE SIP server. The tunnel provides confidentiality, integrity, and data authentication for information that travels across the public network. This occurs using the Secure Real-Time Transport Protocol (SRTP) that has been established using the Session Description Protocol (SDP) and the Security Descriptions for Media Streams (SDES) for SDP - the TOE supports SDES-SRTP.

The SecuSUITE Client also protects communications between itself and the SIP Server by using a Transport Layer Security (TLS)-protected signaling channel. To register the TOE within the domain, the TOE is required to be password authenticated by the SIP Server. The TOE also makes use of certificates to authenticate both the SIP server end and the TOE itself through the TLS connection.

Secure Text Messaging

The SecuSUITE client allows encrypted instant message transfer between client applications. Secure Text Messaging utilizes the same TLS protected SIP communication channel exactly the same way as other sensitive information (such as the SRTP encryption key) is transferred between the clients.

1.4.1 TOE Architecture

The TOE boundary is illustrated in Figure 1-3.

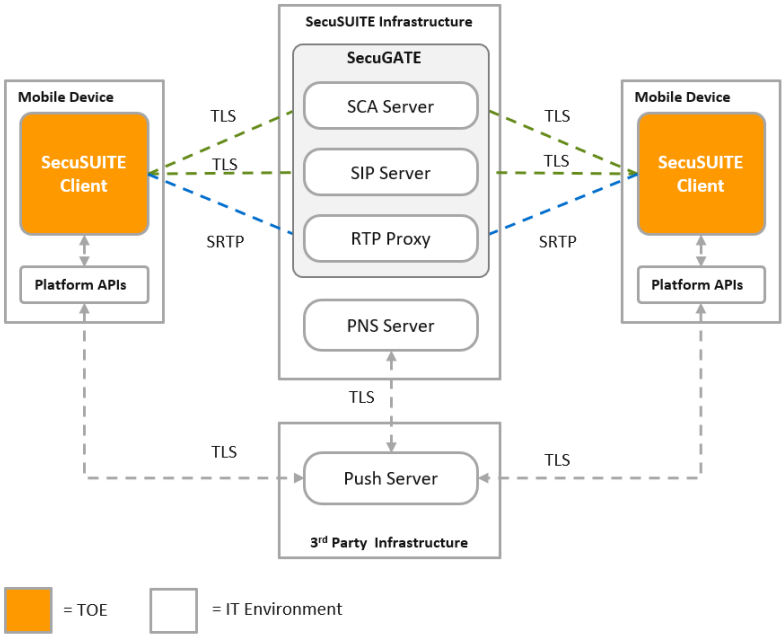

Figure 1-3 TOE Boundary

The TOE is comprised of the SecuSUITE Client software v4.0.

To operate the SecuSUITE Client must be registered to *SecuSUITE* and to the BlackBerry SecuGATE SIP Server. Once properly registered, the Client can initiate or receive a "Call". Call data is exchanged with a VVoIP endpoint through an Enterprise Session Controller (ESC) which provides an SRTP proxy.

Client Registration to SecuSUITE

Before a SecuSUITE client can exchange messages with the SIP Server, it must first be registered to the BlackBerry SecuGATE via the SCA Server. This initial client registration is briefly described below to provide context to the reader:

1. SecuSUITE administrator adds a user to the SecuSUITE via the Admin Portal which generates activation codes that are delivered to the user via some out of band method (e.g. email / printed)
2. User downloads the SecuSUITE client application from supported app store (or it is pushed via an MDM) and launches client app.
3. Client app running on mobile device prompts the user to enter the activation code as well as the SCA Server URL and initiates a TLS connection to SCA Server
4. The user is notified to define a device password in case no device password is defined yet.
5. SCA Server validates client for registration via activation code (this is not the SIP password)
6. Client generates multiple certificate signing requests and submits to SCA Server
7. SCA server's embedded CA creates, signs and returns the certificates
8. Client gets its client configuration settings from SCA server
9. Client gets its SIP settings from SCA server (which retrieves settings from the database server). Settings include:
    a. E.164 telephone number (SIP alias)
    b. SIP Server URI
    c. TLS version (TLS 1.2 only)

d. SIP domain to which client belongs
e. SIP user name and password

10. User performs the following:

a. Enter unique activation code
b. Enter the SCA Server URL

Client Registration with SIP Server

The client registers with the SIP server every time a new connection with the SIP server is established. That is, after:

- Client app was installed and SCA procedure was successfully passed, or
- Client was restarted, or
- Client had lost TLS connection to SIP server (e.g. because of network change or problems)

Procedure:

- Client opens two-way authenticated TLS session with SIP server
- Client registers using SIP REGISTER
- Once registered with the SIP server the client can operate in one of two modes:
  - Constant connection mode, where the client uses periodic requests with the SIP server to keep the TLS connection open.
  - Push Service connection mode where the client registers with a PUSH service on the mobile device and provides that information to the SIP server. When a call is targeted at the client, the SIP server communicates with the PUSH service to awaken the client. Once the mobile device OS wakes the client, the client reconnects to the SIP server to establish a current TLS connection.
- SIP server authenticates client's SIP REGISTER request messages with SIP username and password / digest access authentication.

Digest Access Authentication

The SIP username and password are used to authenticate SIP REGISTER messages using digest access authentication per RFC 3261 as follows:

- Client and server have a shared secret (H(A1) of SIP password)
- Client sends request message to server
- Server rejects request with request message containing challenge ("nonce")
- Client calculates digest from challenge and H(A1) of SIP password
- Client sends request message again with request message now containing digest
- Server also calculates digest and compares this with value received from client
- If digest values match, server accepts request

Call Setup

Preconditions:

- Client A ("Alice") and client B ("Bob") have registered with SCA server.
- Alice and Bob can establish TLS sessions with the SIP server

Alice calls Bob:

- The SIP Server routes SIP messages between Alice and Bob (using a PUSH server to awaken Bob's client if necessary).
- Alice and Bob do not exchange media packets (RTP/RTCP) directly. The SecuSUITE encompasses an RTP proxy which works as an RTP bridge. Alice sends her media packets to the RTP proxy which forwards them to Bob, and vice versa. During connection signaling, the SIP server sets up the RTP/RTCP packet bridging in the RTP proxy for this connection.

The messages are as follows (refer to Annex A: Call Signaling diagram):

- Alice's SIP INVITE message includes:
  - Alice's VoIP Encryption Certificate
- Bob's SIP 200 OK message includes within the SDP:
  - Bob's VoIP Encryption Certificate
  - Bob's SDP message
  - Bob's SRTP master uplink key and salt (i.e. the key and salt Bob is using when sending RTP and RTCP packets to Alice, see (RFC4568, 2006) section 5.1.1) in a message block containing a CMS EnvelopedData ASN.1 structure.
- Alice's SIP ACK includes:
  - Alice's SDP message
  - Alice's SRTP master uplink key and salt (i.e. the key and salt Alice is using when sending RTP and RTCP packets to Bob; similar encoding as Bob)

User Plane (Media)

The main purpose of the RTP Proxy is to make the communication between SIP user agents behind NAT/NAPT possible:

- Typically, the client has an internal (non-routable) IP address and will select some UDP port for RTP and another one for RTCP. NAPT will change the IP address and UDP ports to external values. The internal values however appear in the SDP, and the remote client would use them as destination IP address and ports, which would not work. The solution is to replace the IP address and ports in the SDP: The new IP address is a routable IP address of an RTP proxy, and the RTP/RTCP ports are replaced and used as session identifiers. This replacement happens in the SIP server during call establishment:
- When the SIP server receives the first SIP message with SDP content during call setup (e.g. 200 OK), it extracts the Call-ID, selects an RTP proxy, and sends the Call ID to this RTP Proxy using the RTPproxy Control Protocol.
- The RTP Proxy creates a new session by allocating randomly two subsequent unused UDP ports from a range of UDP ports to that session and returns these port numbers to the SIP server via the RTP Proxy Control Protocol. The first port is for RTP, and the second one for RTCP.
- After receiving the reply from the RTP Proxy, the SIP server replaces the RTP and RTCP media IP addresses and UDP ports in the SDP content of the message with the RTP Proxy IP address and the UDP ports the RTP Proxy has allocated.
- Then the SIP server forwards this modified SIP message as usually to the intended destination.
- When the SIP server receives a SIP follow-up message (e.g. ACK) containing SDP information from the other peer, it sends again the Call-ID to the RTP proxy via the RTPproxy Control Protocol.
- Using the Call-ID as a key, the RTP proxy performs a lookup among existing sessions, allocates randomly another pair of subsequent UDP ports to this session and returns these port numbers to the SIP server.
- After receiving the second pair of port numbers from the RTP proxy, the SIP server replaces the media IP address and Ports in the SDP content of the SIP follow-up message so that it now also points to the RTP proxy. The SIP server forwards the SIP message as usually to the intended destination.
- For RTP, the RTP proxy now listens on the two ports it has allocated for that session and waits for receiving at least one UDP message from Alice and one from Bob. When such a packet is received, the proxy fills one of two IP address/UDP port structures associated to this call with the source IP address and the source UDP port of that packet. When both structures are filled in, the RTP proxy starts relaying UDP/RTP packets between the Alice and Bob.

- The same happens for RTCP.
- The RTP proxy tracks idle time for each of the existing sessions (i.e. the time within which there were no packets relayed), and automatically cleans up sessions whose idle times exceed a specified value (e.g. 60 seconds).

Call Termination

Users can terminate an ongoing call anytime by pushing the "End call" button. The client sends a SIP BYE message and the other party confirms with a SIP OK message. The SIP server then terminates the SRTP session by sending a Delete message for that call to the RTPProxy.

Clients will also terminate a call when no RTP data is received for more than 15 seconds.

1.4.1.1 Physical Boundaries

The TOE executes on the following mobile devices:

a) Samsung Galaxy S9, S9+, S10, S10+, Note9, Note10 (Android 8.0/8.1)

b) Apple iPhone 8, 8 Plus, X, Xs, Xs Max, XR (iOS 12)

Non-TOE Components

The TOE is part of the SecuSUITE security solution and requires the following components to be present in the environment:

a) SecuSUITE SCA Server. The SCA Server authenticates users and facilitates VoIP client enrollment and pushes client SIP configuration to the client.

b) SecuSUITE SIP Server. The SIP Server is used to establish the secure connection between the mobile devices. The use of a TLS connection, providing encryption and mutual authentication, ensures that the devices connect with authorized SIP servers only and the dialed call numbers are transmitted encrypted.

c) SecuSUITE RTP Proxy. The Real-time Transport Protocol (RTP) Proxy bridges media packets sent between clients. The RTP Proxy is part of the SecuSUITE SIP Server. The SIP Server creates and deletes RTP and Real-time Transport Control Protocol (RTCP) bridging sessions in the RTP Proxy.

1.4.1.2 Logical Boundaries

This section summarizes the security functions provided by SecuSUITE Client:

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

1.4.1.2.1 Communication

The TOE utilizes the SILK vocodec to transmit voice media.

1.4.1.2.2 Cryptographic support

The TOE includes its own cryptographic module to perform operations in support of authentication actions and network communications using the TLS and SRTP protocol. The TOE implements TLS version 1.2 with mutual authentication using elliptic-curve cryptography. The TOE also relies upon its platform for certain cryptographic operations including providing random data to seed the TOE's own DRBG. The TOE relies upon the platform (i.e., iOS and Android) cryptographic libraries for operations related to protecting keys in platform offer storage (i.e., a key store).

1.4.1.2.3 User data protection

The TOE enforces the media transmission policy when communicating with remote VVoIP endpoints which use TLS and SRTP protocols. The TOE also ensures that communication with an SCA server is protected using TLS. The TOE protects user data by utilizing platform services for data storage.

1.4.1.2.4 Identification and authentication

The TOE authenticates TLS peers using X.509v3 certificates. It performs extensive X.509 certificate validation checks on these certificates rejecting invalid or revoked certificates.

1.4.1.2.5 Security management

The TOE receives configuration setting during its registration with an SCA server. The client allows management operations that specify the SIP Server to use for connections.

1.4.1.2.6 Privacy

The TOE does not transmit Personally Identifiable Information over any network interfaces.

1.4.1.2.7 Protection of the TSF

The TOE relies on the physical boundary of the evaluated platform as well as the Android and iOS operating systems for the protection of the TOE's application components.

The TOE relies upon these platforms to indicate the current TOE version. If an update is needed, it is obtained from the platform's application store. The TOE's software is digitally signed in accordance with the requirements of each application store.

The native Apple and Android cryptographic library, which provides some of the TOE's cryptographic services, have built-in self-tests that are run at client start-up to ensure that the algorithms are correct. If any self-tests fail, the TOE will not be able to perform its cryptographic services. The TOE includes its own cryptographic library that also includes self-tests that are run when the client starts.

1.4.1.2.8 TOE access

The TOE includes a 15 second default timeout that can terminate idle voice/video transmission. This timeout value can be changed by the configuration obtained from the SCA server.

1.4.1.2.9 Trusted path/channels

The TOE encrypts all data transmitted with an SCA server or Enterprise Session Controller using TLS. The TLS channel established with an ESC can be used to exchange SIP messages or to initiate the use of SRTP for voice/video traffic.

1.4.2 TOE Documentation

BlackBerry Limited offers documentation that describes the installation process for the TOE as well as guidance for subsequent use and administration of the applicable security features of the TOE. The following list of documents were examined as part of the evaluation.

- Common Criteria Configuration Guide BlackBerry SecuSUITE 4.0, Version 1.3, 30-Jan-2020

2. Conformance Claims

This TOE is conformant to the following CC specifications:

- Common Criteria for Information Technology Security Evaluation Part 2: Security functional components, Version 3.1, Revision 5, April 2017.
  - Part 2 Extended
- Common Criteria for Information Technology Security Evaluation Part 3: Security assurance components, Version 3.1, Revision 5, April 2017.
  - Part 3 Conformant
- Package Claims:
  - Protection Profile for Application Software, Version 1.3, 1 March 2019 (ASPP13) with applied technical decisions:

    TD0486 TD0437 TD0416

    TD0465 TD0435

    TD0445 TD0434

    TD0444 TD0427
  - Functional Package for Transport Layer Security (TLS), 1.1, 2019-02-12 (PKGTLS11) with applied technical decisions:

    TD0469

    TD0442
  - Application Software Protection Profile (App PP) Extended Package Voice/Video over IP (VVoIP) Endpoint, Version 1.0, 28 September 2016 (VVoIPASEP10) with applied technical decisions:

    TD0428 TD0372 TD0193

    TD0406 TD0367

    TD0376 TD0279

2.1 Conformance Rationale

The ST conforms to the ASPP13/PKGTLS11/VVoIPASEP10. As explained previously, the security problem definition, security objectives, and security requirements have been drawn from the PP.

3. Security Objectives

The Security Problem Definition may be found in the ASPP13/PKGTLS11/VVoIPASEP10 and this section reproduces only the corresponding Security Objectives for operational environment for reader convenience. The ASPP13/PKGTLS11/VVoIPASEP10 offers additional information about the identified security objectives, but that has not been reproduced here and the ASPP13/PKGTLS11/VVoIPASEP10 should be consulted if there is interest in that material.

In general, the ASPP13/PKGTLS11/VVoIPASEP10 has defined Security Objectives appropriate for software applications that provide Voice/Video over IP (VVoIP) endpoints and as such are applicable to the SecuSUITE Client TOE.

3.1 Security Objectives for the Operational Environment

OE.PLATFORM The TOE relies upon a trustworthy computing platform for its execution. This includes the underlying operating system and any discrete execution environment provided to the TOE.

OE.PROPER_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.

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.

T.MEDIA_DISCLOSURE An attacker can use the encrypted variable rate vocoder frames to their advantage to decode transmitted data.

T.UNDETECTED_TRANSMISSION An attacker may cause the TOE to exfiltrate audio and/or video media over a remote channel while in a state where the user has a reasonable expectation that no media is being transmitted.

4. Extended Components Definition

All of the extended requirements in this ST have been drawn from the ASPP13/PKGTLS11/VVoIPASEP10. The ASPP13/PKGTLS11/VVoIPASEP10 defines the following extended requirements and since they are not redefined in this ST the ASPP13/PKGTLS11/VVoIPASEP10 should be consulted for more information in regard to those CC extensions.

Extended SFRs:

- VVoIPASEP10:FCO_VOC_EXT.1: Fixed-Rate Vocoder
- ASPP13:FCS_CKM_EXT.1: Cryptographic Key Generation Services
- ASPP13:FCS_RBG_EXT.1: Random Bit Generation Services
- ASPP13:FCS_RBG_EXT.2: Random Bit Generation from Application
- VVoIPASEP10:FCS_SRTP_EXT.1: Secure Real-Time Transport Protocol
- ASPP13:FCS_STO_EXT.1: Storage of Credentials
- PKGTLS11:FCS_TLS_EXT.1: TLS Protocol
- PKGTLS11:FCS_TLSC_EXT.1: TLS Client Protocol
- PKGTLS11:FCS_TLSC_EXT.2: TLS Client Support for Mutual Authentication
- PKGTLS11:FCS_TLSC_EXT.5: TLS Client Support for Supported Groups Extension
- ASPP13:FDP_DAR_EXT.1: Encryption Of Sensitive Application Data
- ASPP13:FDP_DEC_EXT.1: Access to Platform Resources
- ASPP13:FDP_NET_EXT.1: Network Communications
- ASPP13:FIA_X509_EXT.1: X.509 Certificate Validation
- ASPP13:FIA_X509_EXT.2: X.509 Certificate Authentication
- ASPP13:FMT_CFG_EXT.1: Secure by Default Configuration
- ASPP13:FMT_MEC_EXT.1: Supported Configuration Mechanism
- ASPP13:FPR_ANO_EXT.1: User Consent for Transmission of Personally Identifiable
- ASPP13:FPT_AEX_EXT.1: Anti-Exploitation Capabilities
- ASPP13:FPT_API_EXT.1: Use of Supported Services and APIs
- ASPP13:FPT_IDV_EXT.1: Software Identification and Versions
- ASPP13:FPT_LIB_EXT.1: Use of Third Party Libraries
- ASPP13:FPT_TUD_EXT.1: Integrity for Installation and Update
- ASPP13:FPT_TUD_EXT.2: Integrity for Installation and Update
- ASPP13:FTP_DIT_EXT.1: Protection of Data in Transit
- VVoIPASEP10:FTP_DIT_EXT.1: Protection of Data in Transit

Extended SARs:

- - ALC_TSU_EXT.1: Timely Security Updates

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 Target of Evaluation (TOE) and to scope the evaluation effort.

The SFRs have all been drawn from the ASPP13/PKGTLS11/VVoIPASEP10. The refinements and operations already performed in the ASPP13/PKGTLS11/VVoIPASEP10 are not identified (e.g., highlighted) here, rather the requirements have been copied from the ASPP13/PKGTLS11/VVoIPASEP10 and any residual operations have been completed herein. Of particular note, the ASPP13/PKGTLS11/VVoIPASEP10 made a number of refinements and completed some of the SFR operations defined in the Common Criteria (CC) and that PP should be consulted to identify those changes if necessary.

The SARs are also drawn from the ASPP13/PKGTLS11/VVoIPASEP10 which includes all the SARs for EAL 1. However, the SARs are effectively refined since requirement-specific 'Assurance Activities' are defined in the ASPP13/PKGTLS11/VVoIPASEP10 that serve to ensure corresponding evaluations will yield more practical and consistent assurance than the EAL 1 assurance requirements alone. The ASPP13/PKGTLS11/VVoIPASEP10 should be consulted for the assurance activity definitions.

5.1 TOE Security Functional Requirements

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

Requirement Class	Requirement Component
FCO: Communication	VVoIPASEP10:FCO_VOC_EXT.1: Fixed-Rate Vocoder
FCS: Cryptographic support	ASPP13:FCS_CKM.1(1): Cryptographic Asymmetric Key Generation
	ASPP13:FCS_CKM.2: Cryptographic Key Establishment
	ASPP13:FCS_CKM_EXT.1: Cryptographic Key Generation Services
	ASPP13:FCS_COP.1(1): Cryptographic Operation - Encryption/Decryption
	ASPP13:FCS_COP.1(2): Cryptographic Operation - Hashing
	ASPP13:FCS_COP.1(3): Cryptographic Operation - Signing
	ASPP13:FCS_COP.1(4): Cryptographic Operation - Keyed-Hash Message Authentication
	VVoIPASEP10:FCS_COP.1(5): Cryptographic Operation - Encryption/Decryption for SRTP
	ASPP13:FCS_RBG_EXT.1: Random Bit Generation Services
	ASPP13:FCS_RBG_EXT.2: Random Bit Generation from Application
	VVoIPASEP10:FCS_SRTP_EXT.1: Secure Real-Time Transport Protocol
	ASPP13:FCS_STO_EXT.1: Storage of Credentials
	PKGTLS11:FCS_TLS_EXT.1: TLS Protocol
	PKGTLS11:FCS_TLSC_EXT.1: TLS Client Protocol
	PKGTLS11:FCS_TLSC_EXT.2: TLS Client Support for Mutual Authentication
	PKGTLS11:FCS_TLSC_EXT.5: TLS Client Support for Supported Groups Extension
FDP: User data protection	ASPP13:FDP_DAR_EXT.1: Encryption Of Sensitive Application Data
	ASPP13:FDP_DEC_EXT.1: Access to Platform Resources
	VVoIPASEP10:FDP_IFC.1: Subset Information Flow Control
	VVoIPASEP10:FDP_IFF.1: Information Flow Control Functions

	ASPP13:FDP_NET_EXT.1: Network Communications
FIA: Identification and authentication	ASPP13:FIA_X509_EXT.1: X.509 Certificate Validation
	ASPP13:FIA_X509_EXT.2: X.509 Certificate Authentication
FMT: Security management	ASPP13:FMT_CFG_EXT.1: Secure by Default Configuration
	ASPP13:FMT_MEC_EXT.1: Supported Configuration Mechanism
	ASPP13:FMT_SMF.1: Specification of Management Functions
	VVoIPASEP10:FMT_SMF.1: Specification of Management Functions
FPR: Privacy	ASPP13:FPR_ANO_EXT.1: User Consent for Transmission of Personally Identifiable
FPT: Protection of the TSF	ASPP13:FPT_AEX_EXT.1: Anti-Exploitation Capabilities
	ASPP13:FPT_API_EXT.1: Use of Supported Services and APIs
	ASPP13:FPT_IDV_EXT.1: Software Identification and Versions
	ASPP13:FPT_LIB_EXT.1: Use of Third Party Libraries
	ASPP13:FPT_TUD_EXT.1: Integrity for Installation and Update
	ASPP13:FPT_TUD_EXT.2: Integrity for Installation and Update
FTA: TOE access	VVoIPASEP10:FTA_SSL.3(1): TSF-Initiated Termination (Media Channel)
FTP: Trusted path/channels	ASPP13:FTP_DIT_EXT.1: Protection of Data in Transit
	VVoIPASEP10:FTP_DIT_EXT.1: Protection of Data in Transit
	VVoIPASEP10:FTP_ITC.1/Control: Inter-TSF Trusted Channel (Signaling Channel) - Control
	VVoIPASEP10:FTP_ITC.1/Media: Inter-TSF Trusted Channel (Media Channel)

Table 5-1 TOE Security Functional Components

5.1.1 Communication (FCO)

5.1.1.1 Fixed-Rate Vocoder (VVoIPASEP10:FCO_VOC_EXT.1)

VVoIPASEP10:FCO_VOC_EXT.1.1

The TSF shall transmit voice media using a constant bit rate voice vocoder.

5.1.2 Cryptographic support (FCS)

5.1.2.1 Cryptographic Asymmetric Key Generation (ASPP13:FCS_CKM.1(1))

ASPP13:FCS_CKM.1.1(1)

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-256, P-384 and [no other curves] that meet the following: FIPS PUB 186-4, 'Digital Signature Standard (DSS)', Appendix B.4***].

5.1.2.2 Cryptographic Key Establishment (ASPP13:FCS_CKM.2)

ASPP13:FCS_CKM.2.1

The application shall [***implement functionality***] to perform cryptographic key establishment in accordance with a specified cryptographic key establishment method: [***Elliptic curve-based key establishment schemes that meets the following: NIST Special Publication 800-56A,***

'Recommendation for Pair-Wise Key Establishment Schemes Using Discrete Logarithm Cryptography'].

5.1.2.3 Cryptographic Key Generation Services (ASPP13:FCS_CKM_EXT.1)

ASPP13:FCS_CKM_EXT.1.1

The application shall [***invoke platform-provided functionality for asymmetric key generation (iOS implementation only), implement asymmetric key generation (Android implementation only)***].

Application Note: The Android application will implement this functionality. The iOS application will invoke platform provided functionality.

5.1.2.4 Cryptographic Operation - Encryption/Decryption (ASPP13:FCS_COP.1(1))

ASPP13:FCS_COP.1.1(1)

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 cryptographic key sizes [***256-bit***].

5.1.2.5 Cryptographic Operation - Hashing (ASPP13:FCS_COP.1(2))

ASPP13:FCS_COP.1.1(2)

The application shall perform cryptographic hashing services in accordance with a specified cryptographic algorithm [***SHA-1, SHA-256, SHA-384, SHA-512***] and message digest sizes [***160, 256, 384, 512***] bits that meet the following: FIPS Pub 180-4.

5.1.2.6 Cryptographic Operation - Signing (ASPP13:FCS_COP.1(3))

ASPP13:FCS_COP.1.1(3)

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 186-4, 'Digital Signature Standard (DSS)', Section 4,***
- ***ECDSA schemes using 'NIST curves' P-256, P-384 and [no other curves] that meet the following: FIPS PUB 186-4, 'Digital Signature Standard (DSS)', Section 5***].

5.1.2.7 Cryptographic Operation - Keyed-Hash Message Authentication (ASPP13:FCS_COP.1(4))

ASPP13:FCS_COP.1.1(4)

The application shall perform keyed-hash message authentication in accordance with a specified cryptographic algorithm HMAC-SHA-256 and [***SHA-1, SHA-384***] with key sizes [***160, 256, 384***] and message digest sizes 256 and [***160, 384***] bits that meet the following: FIPS Pub 198-1 The Keyed-Hash Message Authentication Code and FIPS Pub 180-4 Secure Hash Standard.

5.1.2.8 Cryptographic Operation - Encryption/Decryption for SRTP (VVoIPASEP10:FCS_COP.1(5))

VVoIPASEP10:FCS_COP.1.1(5)

Refinement: The application shall perform encryption/decryption to support SDES-SRTP in accordance with a specified cryptographic algorithm AES-CTR (as defined in NIST SP 800-38A) mode; and [***no other modes***] and cryptographic key sizes 128-bit and [***256-bit***]. (Per TD0193)

5.1.2.9 Random Bit Generation Services (ASPP13:FCS_RBG_EXT.1)

ASPP13:FCS_RBG_EXT.1.1

The application shall [***implement DRBG functionality***] for its cryptographic operations.

5.1.2.10 Random Bit Generation from Application (ASPP13:FCS_RBG_EXT.2)

ASPP13: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)***].

ASPP13:FCS_RBG_EXT.2.2
The deterministic RBG shall be seeded by an entropy source that accumulates entropy from a platform-based DRBG and [***no other noise source***] with a minimum of [***256 bits***] 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.

5.1.2.11 Secure Real-Time Transport Protocol (VVoIPASEP10:FCS_SRTP_EXT.1)

VVoIPASEP10:FCS_SRTP_EXT.1.1
The TSF shall implement the Secure Real-Time Transport Protocol (SRTP) that complies with RFC 3711, and use Security Descriptions for Media Streams (SDES) in compliance with RFC 4568 to provide key information for the SRTP connection.

VVoIPASEP10:FCS_SRTP_EXT.1.2
The TSF shall implement SDES-SRTP supporting the following ciphersuites:
[***AES_256_CM_HMAC_SHA1_80, in accordance with RFC 6188***].
(TD0279 applied)

VVoIPASEP10:FCS_SRTP_EXT.1.3
The TSF shall ensure the SRTP NULL algorithm can be disabled.

VVoIPASEP10:FCS_SRTP_EXT.1.4
The TSF shall allow the SRTP ports to be used for SRTP communications to be specified by an Authorized Administrator.

5.1.2.12 Storage of Credentials (ASPP13:FCS_STO_EXT.1)

ASPP13:FCS_STO_EXT.1.1
The application shall [***invoke the functionality provided by the platform to securely store [secret and private keys]***] to non-volatile memory.

5.1.2.13 TLS Protocol (PKGTLS11:FCS_TLS_EXT.1)

PKGTLS11:FCS_TLS_EXT.1.1
The product shall implement [***TLS as a client,***]

5.1.2.14 TLS Client Protocol (PKGTLS11:FCS_TLSC_EXT.1)

PKGTLS11:FCS_TLSC_EXT.1.1
The product shall implement TLS 1.2 (RFC 5246) and [***no earlier TLS versions***] as a client that supports the cipher suites [
TLS_ECDHE_ECDSA_WITH_AES_256_GCM_SHA384 as defined in RFC 5289,
TLS_ECDHE_RSA_WITH_AES_256_GCM_SHA384 as defined in RFC5289]
and also supports functionality for [***mutual authentication***]

PKGTLS11:FCS_TLSC_EXT.1.2
The product shall verify that the presented identifier matches the reference identifier according to RFC 6125.

PKGTLS11:FCS_TLSC_EXT.1.3
The product shall not establish a trusted channel if the server certificate is invalid [***with no exceptions***]

5.1.2.15 TLS Client Support for Mutual Authentication (PKGTLS11:FCS_TLSC_EXT.2)

PKGTLS11:FCS_TLSC_EXT.2.1
The product shall support mutual authentication using X.509v3 certificates.

5.1.2.16 TLS Client Support for Supported Groups Extension (PKGTLS11:FCS_TLSC_EXT.5)

PKGTLS11:FCS_TLSC_EXT.5.1
The product shall present the Supported Groups Extension in the Client Hello with the supported groups [***secp256r1, secp384r1***]

5.1.3 User data protection (FDP)

5.1.3.1 Encryption Of Sensitive Application Data (ASPP13:FDP_DAR_EXT.1)

ASPP13:FDP_DAR_EXT.1.1
The application shall [***protect sensitive data in accordance with FCS_STO_EXT.1***] in non-volatile memory.

5.1.3.2 Access to Platform Resources (ASPP13:FDP_DEC_EXT.1)

ASPP13:FDP_DEC_EXT.1.1
The application shall restrict its access to [***network connectivity, camera, microphone, Bluetooth, [proximity sensor, notifications]***].

ASPP13:FDP_DEC_EXT.1.2
The application shall restrict its access to [***address book***].

5.1.3.3 Subset Information Flow Control (VVoIPASEP10:FDP_IFC.1)

VVoIPASEP10:FDP_IFC.1.1
The TSF shall enforce the media transmission policy on voice/video media transmitted by the TOE.

5.1.3.4 Information Flow Control Functions (VVoIPASEP10:FDP_IFF.1)

VVoIPASEP10:FDP_IFF.1.1
The TSF shall enforce the media transmission policy based on the following types of subject and information security attributes: ESC registration status and TOE hook state.

VVoIPASEP10:FDP_IFF.1.2
The TSF shall permit an information flow between a controlled subject and controlled information via a controlled operation if the following rules hold:

- The TOE is registered with the ESC,
- A call has been established with a telephony device (VVoIP endpoint),
- The TOE is in the off-hook state,
- The TOE is not in the mute state,
- [***No other rules***].

VVoIPASEP10:FDP_IFF.1.3
The TSF shall enforce the no additional information flow control policy rules.

VVoIPASEP10:FDP_IFF.1.4
The TSF shall explicitly authorize an information flow based on the following rules: no additional rules.

VVoIPASEP10:FDP_IFF.1.5
The TSF shall explicitly deny an information flow based on the following rules: all TCP and UDP ports used by the TOE are closed when not in active use.

5.1.3.5 Network Communications (ASPP13:FDP_NET_EXT.1)

ASPP13:FDP_NET_EXT.1.1

The application shall restrict network communication to [***user-initiated communication for [registration of the client with an SCA, initiation of an outgoing call, and sending encrypted instant messages], respond to [an incoming call, receipt of an encrypted instant message]***].

5.1.4 Identification and authentication (FIA)

5.1.4.1 X.509 Certificate Validation (ASPP13:FIA_X509_EXT.1)

ASPP13: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 and that the CA flag is set to TRUE for all CA certificates.
- The application shall validate the revocation status of the certificate using [***a Certificate Revocation List (CRL) as specified in RFC 5759***].
- The application shall validate the extendedKeyUsage field according to the following rules:
  - Certificates used for trusted updates and executable code integrity verification shall have the Code Signing purpose (id-kp 3 with OID 1.3.6.1.5.5.7.3.3) in the extendedKeyUsage field.
  - Server certificates presented for TLS shall have the Server Authentication purpose (id-kp 1 with OID 1.3.6.1.5.5.7.3.1) in the extendedKeyUsage field.
  - Client certificates presented for TLS shall have the Client Authentication purpose (id-kp 2 with OID 1.3.6.1.5.5.7.3.2) in the extendedKeyUsage field.
  - S/MIME certificates presented for email encryption and signature shall have the Email Protection purpose (id-kp 4 with OID 1.3.6.1.5.5.7.3.4) in the extendedKeyUsage field.
  - OCSP certificates presented for OCSP responses shall have the OCSP Signing purpose (id-kp 9 with OID 1.3.6.1.5.5.7.3.9) in the extendedKeyUsage field.
  - Server certificates presented for EST shall have the CMC Registration Authority (RA) purpose (id-kpcmcRA with OID 1.3.6.1.5.5.7.3.28) in the extendedKeyUsage field.

ASPP13: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.

5.1.4.2 X.509 Certificate Authentication (ASPP13:FIA_X509_EXT.2)

ASPP13:FIA_X509_EXT.2.1

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

ASPP13:FIA_X509_EXT.2.2

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

5.1.5 Security management (FMT)

5.1.5.1 Secure by Default Configuration (ASPP13:FMT_CFG_EXT.1)

ASPP13:FMT_CFG_EXT.1.1
The application shall provide only enough functionality to set new credentials when configured with default credentials or no credentials.

ASPP13:FMT_CFG_EXT.1.2
The application shall be configured by default with file permissions which protect the application's binaries and data files from modification by normal unprivileged user.

5.1.5.2 Supported Configuration Mechanism (ASPP13:FMT_MEC_EXT.1)

ASPP13:FMT_MEC_EXT.1.1
The application shall [***invoke the mechanisms recommended by the platform vendor for storing and setting configuration options***] (TD0437 applied).

5.1.5.3 Specification of Management Functions (ASPP13:FMT_SMF.1)

ASPP13:FMT_SMF.1.1
The TSF shall be capable of performing the following management functions [***
*1) Specify the SIP Server to use for connections. ***].

5.1.5.4 Specification of Management Functions (VVoIPASEP10:FMT_SMF.1)

VVoIPASEP10:FMT_SMF.1.1
The TSF shall be capable of performing the following management functions: [***
- ***Specify the SIP Server to use for connections. ***]

5.1.6 Privacy (FPR)

5.1.6.1 User Consent for Transmission of Personally Identifiable (ASPP13:FPR_ANO_EXT.1)

ASPP13:FPR_ANO_EXT.1.1
The application shall [***not transmit PII over a network***].

5.1.7 Protection of the TSF (FPT)

5.1.7.1 Anti-Exploitation Capabilities (ASPP13:FPT_AEX_EXT.1)

ASPP13:FPT_AEX_EXT.1.1
The application shall not request to map memory at an explicit address except for [***none***].

ASPP13:FPT_AEX_EXT.1.2
The application shall [***not allocate any memory region with both write and execute permissions***].

ASPP13:FPT_AEX_EXT.1.3
The application shall be compatible with security features provided by the platform vendor.

ASPP13: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.

ASPP13:FPT_AEX_EXT.1.5
The application shall be built with stack-based buffer overflow protection enabled.

5.1.7.2 Use of Supported Services and APIs (ASPP13:FPT_API_EXT.1)

ASPP13:FPT_API_EXT.1.1
The application shall use only documented platform APIs.

5.1.7.3 Software Identification and Versions (ASPP13:FPT_IDV_EXT.1)

ASPP13:FPT_IDV_EXT.1.1
The application shall be versioned with [***[a multi-part unique release number]***]

5.1.7.4 Use of Third Party Libraries (ASPP13:FPT_LIB_EXT.1)

ASPP13:FPT_LIB_EXT.1.1
The application shall be packaged with only [***libraries shown in*** **Table 5-2**].

Android & iOS	Android Only
• webrtc 1.0.0 • Boost 1.67.0 • ZLIB 1.2.11 • BZip2 1.0.8 • Openssl 1.0.2r • pjproject 2.7.1 • PocoCpp 1.9.1 • libphonenumber 7.7.3 • opus 1.1.3 • silk 1.0.9 • libsrtp 1.5.4	• joda-time 2.1 • spongycastle 1.54.0.0 • androidannotations 4.6.0 • firebase 20.0.0 • androidx.appcompat 1.1.0 • zxing barcode scanner 3.6.0 • kotlin-stdlib-jdk7 1.3.50 • androidx.biometric 1.0.0-beta01

Table 5-2 Third Party Libraries in TOE

5.1.7.5 Integrity for Installation and Update (ASPP13:FPT_TUD_EXT.1)

ASPP13:FPT_TUD_EXT.1.1
The application shall [***leverage the platform***] to check for updates and patches to the application software.

ASPP13:FPT_TUD_EXT.1.2
The application shall [***leverage the platform***] to query the current version of the application software.

ASPP13:FPT_TUD_EXT.1.3
The application shall not download, modify, replace or update its own binary code.

ASPP13:FPT_TUD_EXT.1.4
The application installation package and its updates shall be digitally signed such that its platform can cryptographically verify them prior to installation

ASPP13:FPT_TUD_EXT.1.5
The application is distributed [***as an additional software package to the platform OS***].

5.1.7.6 Integrity for Installation and Update (ASPP13:FPT_TUD_EXT.2)

ASPP13:FPT_TUD_EXT.2.1
The application shall be distributed using the format of the platform-supported package manager.

ASPP13: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

5.1.7.7 Integrity for Installation and Update (VVoIPASEP10:FPT_TUD_EXT.1)

Removed per TD0367

5.1.8 TOE access (FTA)

5.1.8.1 TSF-Initiated Termination (Media Channel) (VVoIPASEP10:FTA_SSL.3(1))

VVoIPASEP10:FTA_SSL.3(1).1
The TSF shall terminate voice/video transmission after [***15***] seconds, an administrator-configurable interval on the [***ESC***] downloaded to the TOE during configuration.

5.1.9 Trusted path/channels (FTP)

5.1.9.1 Protection of Data in Transit (ASPP13:FTP_DIT_EXT.1)

ASPP13:FTP_DIT_EXT.1.1
The application shall [***encrypt all transmitted [data] with [TLS as defined in the TLS Package]***] between itself and another trusted IT product.

5.1.9.2 Protection of Data in Transit (VVoIPASEP10:FTP_DIT_EXT.1)

VVoIPASEP10:FTP_DIT_EXT.1.1
The application shall [encrypt all transmitted data with TLS, [***SRTP***] between itself and another trusted IT product.

5.1.9.3 Inter-TSF Trusted Channel (Signaling Channel) - Control (VVoIPASEP10:FTP_ITC.1/Control)

VVoIPASEP10:FTP_ITC.1.1/Control
The TSF shall be capable of using [***SIP***] to provide a trusted communication channel between itself and an Enterprise Session Controller that is logically distinct from other communication channels and provides assured identification of its end points and protection of the channel data from disclosure and detection of modification of the channel data.

VVoIPASEP10:FTP_ITC.1.2/Control
The TSF shall permit the [***TSF, the Enterprise Session Controller***] to initiate communication via the trusted channel.

VVoIPASEP10:FTP_ITC.1.3/Control
The TSF shall initiate communication via the trusted channel for [*establishment of call control*].

5.1.9.4 Inter-TSF Trusted Channel (Media Channel) (VVoIPASEP10:FTP_ITC.1/Media)

VVoIPASEP10:FTP_ITC.1.1/Media
The TSF shall be capable of using [***SRTP***] to provide a trusted communication channel between itself and another VVoIP endpoint or other telephony device that is logically distinct from other communication channels and provides assured identification of its end points and protection of the channel data from disclosure and detection of modification of the channel data.

VVoIPASEP10:FTP_ITC.1.2/Media
The TSF shall permit the [***TSF, another VVoIP endpoint or other telephony device***] to initiate communication via the trusted channel.

VVoIPASEP10:FTP_ITC.1.3/Media
The TSF shall initiate communication via the trusted channel for [*transmission of voice/video media*].

5.2 TOE Security Assurance Requirements

The SARs for the TOE are the components as specified in Part 3 of the Common Criteria. Note that the SARs have effectively been refined with the assurance activities explicitly defined in association with both the SFRs and SARs.

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 â€“ Conformance
AVA: Vulnerability assessment	AVA_VAN.1: Vulnerability Survey

Table 5-3 Assurance Components

5.2.1 Development (ADV)

5.2.1.1 Basic Functional Specification (ADV_FSP.1)

ADV_FSP.1.1d
The developer shall provide a functional specification.

ADV_FSP.1.2d
The developer shall provide a tracing from the functional specification to the SFRs.

ADV_FSP.1.1c
The functional specification shall describe the purpose and method of use for each SFR-enforcing and SFR-supporting TSFI.

ADV_FSP.1.2c
The functional specification shall identify all parameters associated with each SFR-enforcing and SFR-supporting TSFI.

ADV_FSP.1.3c
The functional specification shall provide rationale for the implicit categorization of interfaces as SFR-non-interfering.

ADV_FSP.1.4c
The tracing shall demonstrate that the SFRs trace to TSFIs in the functional specification.

ADV_FSP.1.1e
The evaluator shall confirm that the information provided meets all requirements for content and presentation of evidence.

ADV_FSP.1.2e
The evaluator shall determine that the functional specification is an accurate and complete instantiation of the SFRs.

5.2.2 Guidance documents (AGD)

5.2.2.1 Operational User Guidance (AGD_OPE.1)

AGD_OPE.1.1d
The developer shall provide operational user guidance.

AGD_OPE.1.1c

The operational user guidance shall describe, for each user role, the user accessible functions and privileges that should be controlled in a secure processing environment, including appropriate warnings.

AGD_OPE.1.2c

The operational user guidance shall describe, for each user role, how to use the available interfaces provided by the TOE in a secure manner.

AGD_OPE.1.3c

The operational user guidance shall describe, for each user role, the available functions and interfaces, in particular all security parameters under the control of the user, indicating secure values as appropriate.

AGD_OPE.1.4c

The operational user guidance shall, for each user role, clearly present each type of security-relevant event relative to the user-accessible functions that need to be performed, including changing the security characteristics of entities under the control of the TSF.

AGD_OPE.1.5c

The operational user guidance shall identify all possible modes of operation of the TOE (including operation following failure or operational error), their consequences, and implications for maintaining secure operation.

AGD_OPE.1.6c

The operational user guidance shall, for each user role, describe the security measures to be followed in order to fulfill the security objectives for the operational environment as described in the ST.

AGD_OPE.1.7c

The operational user guidance shall be clear and reasonable.

AGD_OPE.1.1e

The evaluator shall confirm that the information provided meets all requirements for content and presentation of evidence.

5.2.2.2 Preparative Procedures (AGD_PRE.1)

AGD_PRE.1.1d

The developer shall provide the TOE, including its preparative procedures.

AGD_PRE.1.1c

The preparative procedures shall describe all the steps necessary for secure acceptance of the delivered TOE in accordance with the developer's delivery procedures.

AGD_PRE.1.2c

The preparative procedures shall describe all the steps necessary for secure installation of the TOE and for the secure preparation of the operational environment in accordance with the security objectives for the operational environment as described in the ST.

AGD_PRE.1.1e

The evaluator shall confirm that the information provided meets all requirements for content and presentation of evidence.

AGD_PRE.1.2e

The evaluator shall apply the preparative procedures to confirm that the TOE can be prepared securely for operation.

5.2.3 Life-cycle support (ALC)

5.2.3.1 Labelling of the TOE (ALC_CMC.1)

ALC_CMC.1.1d

The developer shall provide the TOE and a reference for the TOE.

ALC_CMC.1.1c
The TOE shall be labelled with its unique reference.
ALC_CMC.1.1e
The evaluator shall confirm that the information provided meets all requirements for content and presentation of evidence.

5.2.3.2 TOE CM Coverage (ALC_CMS.1)

ALC_CMS.1.1d
The developer shall provide a configuration list for the TOE.
ALC_CMS.1.1c
The configuration list shall include the following: the TOE itself; and the evaluation evidence required by the SARs.
ALC_CMS.1.2c
The configuration list shall uniquely identify the configuration items.
ALC_CMS.1.1e
The evaluator shall confirm that the information provided meets all requirements for content and presentation of evidence.

5.2.3.3 Timely Security Updates (ALC_TSU_EXT.1)

ALC_TSU_EXT.1.1d
The developer shall provide a description in the TSS of how timely security updates are made to the TOE. Note: Application developers must support updates to their products for purposes of fixing security vulnerabilities.
ALC_TSU_EXT.1.2d
The developer shall provide a description in the TSS of how users are notified when updates change security properties or the configuration of the product.
ALC_TSU_EXT.1.1c
The description shall include the process for creating and deploying security updates for the TOE software.
ALC_TSU_EXT.1.2c
The description shall express the time window as the length of time, in days, between public disclosure of a vulnerability and the public availability of security updates to the TOE.
ALC_TSU_EXT.1.3c
The description shall include the mechanisms publicly available for reporting security issues pertaining to the TOE. Note: The reporting mechanism could include web sites, email addresses, as well as a means to protect the sensitive nature of the report (e.g., public keys that could be used to encrypt the details of a proof-of-concept exploit).
ALC_TSU_EXT.1.1e
The evaluator shall confirm that the information provided meets all requirements for content and presentation of evidence.

5.2.4 Tests (ATE)

5.2.4.1 Independent Testing Conformance (ATE_IND.1)

ATE_IND.1.1d
The developer shall provide the TOE for testing.
ATE_IND.1.1c
The TOE shall be suitable for testing.
ATE_IND.1.1e
The evaluator shall confirm that the information provided meets all requirements for content and presentation of evidence.

ATE_IND.1.2e

The evaluator shall test a subset of the TSF to confirm that the TSF operates as specified.

5.2.5 Vulnerability assessment (AVA)

5.2.5.1 Vulnerability Survey (AVA_VAN.1)

AVA_VAN.1.1d

The developer shall provide the TOE for testing.

AVA_VAN.1.1c

The TOE shall be suitable for testing.

AVA_VAN.1.1e

The evaluator shall confirm that the information provided meets all requirements for content and presentation of evidence.

AVA_VAN.1.2e

The evaluator shall perform a search of public domain sources to identify potential vulnerabilities in the TOE.

AVA_VAN.1.3e

The evaluator shall conduct penetration testing, based on the identified potential vulnerabilities, to determine that the TOE is resistant to attacks performed by an attacker possessing Basic attack potential.

6. TOE Summary Specification

This chapter describes the security functions:

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

6.1 Communication

The Communication function satisfies the following security functional requirements:

- VVoIPASEP10:FCO_VOC_EXT.1: The TOE uses the SILK vocoder with a padding. The combination of the codec and padding ensures that the bit-rate is constant throughout the duration of the call.

6.2 Cryptographic support

The TOE leverages the client device platform RBG to seed the OpenSSL CTR_DRBG (AES). The client device platform RBG functionality is invoked as follows.

- On Samsung devices, random data is read by invoking the OpenSSL RAND API; or
- On Apple devices, random data is obtained by invoking SecRandomCopyBytes that reads random data from /dev/random.

The CAVP certificates shown as C1422 in **Table 6-1 Cryptographic Functions** were obtained during the evaluation testing.

A11 Bionic – iPhone 8, 8 Plus, X

A12 Bionic – iPhone Xs, Xs Max, XR

Exynos 9810 – Galaxy S9, S9+, Note9

Exynos 9820 – Galaxy S10, S10+, Note10

SDM845 – Galaxy S9, S9+, Note9

SDM855 – Galaxy S10, S10+, Note10

Functions	Requirement	Cert #
Encryption/Decryption		
AES CBC (256 bits) AES GCM (256 bits)	ASPP13:FCS_COP.1(1)	C1422
Cryptographic hashing		
SHA-1, SHA-256, SHA-384, SHA-512 (digest sizes 160, 256, 384, 512)	ASPP13:FCS_COP.1(2)	C1422
Cryptographic signature services		
Elliptic Curve Digital Signature Algorithm (ECDSA) with an elliptical curve size of 256 or 384 bits	ASPP13:FCS_COP.1(3)	C1422

RSA Digital Signature with 2048 or 3072 bit keys (verification only)		
Keyed-hash message authentication		
HMAC-SHA-256, HMAC-SHA-1, HMAC-SHA-384 (digest sizes 160, 256, 384)	ASPP13:FCS_COP.1(4)	C1422
Encryption/Decryption for SRTP		
AES-CTR (256 bit) AES-GCM (256-bit)	VVoIPASEP10:FCS_COP.1.1(5)	C1422
Random bit generation		
CTR_DRBG with sw based noise sources with a minimum of 256 bits of non-determinism	ASPP13:FCS_RBC_EXT.1	C1422
Asymmetric Key Generation		
ECDSA Key Generation	ASPP13:FCS_CKM.1	C1422
Key Establishment		
Elliptic curve-based key establishment schemes	ASPP13:FCS_CKM.2	C1422

Table 6-1 Cryptographic Functions

The TOE generates ECDSA P-384 keys to generate certificates as well as for TLS key exchange. The TOE generates ECDSA P-256 keys for backwards TLS key exchange compatibility with older BlackBerry SecuGATE SIP servers.

The TOE supports SRTP protocol as described by RFC 3711 using Security Descriptions for Media Streams (SDES) in compliance with RFC 4568. The TOE implements the AES_256_CM_HMAC_SHA1_80 ciphersuite for SDES-SRTP and does not allow the NULL algorithm to be specified.

The TOE uses SHA-256 as the SIP message digest authentication mechanism during a SIP transaction. All SIP messages are protected via TLS.

The SRTP destination ports that shall be used for a specific call are ‘negotiated’ between RTP Proxy and SIP Server and included into the SDES body of the forwarded SIP messages. The admin can restrict the port range in the SIP server configuration.

The TOE supports the following ciphersuites with only TLS v1.2 communication:

- TLS_ECDHE_ECDSA_WITH_AES_256_GCM_SHA384
- TLS_ECDHE_RSA_WITH_AES_256_GCM_SHA384

All older versions of TLS are rejected. The TOE also support mutual authentication during a TLS negotiation.

The Cryptographic support function satisfies the following security functional requirements:

- ASPP13:FCS_CKM.1(1): The TOE supports asymmetric key generation using the Openssl cryptographic functions described above.
- ASPP13:FCS_CKM.2: The TOE supports key establishment using Elliptic curve-based schemes described above.
- ASPP13:FCS_CKM_EXT.1: For an Android platform, the TOE uses its Openssl implementation to generate keys. For the iOS platform, the TOE uses platform provided API to generate keys.
- ASPP13:FCS_COP.1(1): The TOE performs encryption and decryption operations using the Openssl AES cryptographic functions described above.
- ASPP13:FCS_COP.1(2): The TOE performs hashing operations using the Openssl SHA cryptographic functions described above.

- ASPP13:FCS_COP.1(3): The TOE performs signature operations using the Openssl cryptographic functions described above.

  The SecuSUITE TOE performs RSA signature verification only. The TOE only has an ECDSA certificate for mutual authentication. The TOE can connect to a SecuGATE server that might be configured to send an RSA server certificate to the TOE. In this case, the TOE can only verify the RSA certificate signature, and the TOE will send an ECDSA certificate to the server during mutual authentication.

- ASPP13:FCS_COP.1(4): The TOE performs signature operations using the Openssl cryptographic functions described above.
- VVoIPASEP10:FCS_COP.1(5): The TOE supports SDES-SRTP as described above
- ASPP13:FCS_RBG_EXT.1: The TOE leverages platform RBG services to seed its own OpenSSL CTR_DRBG(AES) as described above.
- ASPP13:FCS_RBG_EXT.2: The TOE implements its own CTR_DRBG(AES) which is seeded from platform RBG services with a minimum of 256 bits of entropy.
- VVoIPASEP10:FCS_SRTP_EXT.1: The TOE supports SRTP as described above.
- ASPP13:FCS_STO_EXT.1: The TOE stores X509v3 certificate private keys persistently. The TOE uses X509 certificates to authenticate to the SIP server (ESC) when performing TLS. The TOE stores the data when not in use in client device platform-provided key storage as follows:
  - Samsung. AndroidKeyStore.
  - Apple. iOS keychain.
- PKGTLS11:FCS_TLS_EXT.1: The TOE acts as a TLS client.
- PKGTLS11:FCS_TLSC_EXT.1: The TOE supports TLS communication with mutual authentication as described above. The TOE uses the connection URL as the reference identifier to compare against the identifier in the peer's certificate. The TOE only supports SAN reference identifier checking. The TOE verifies the certificate received matches the reference identifier for the expected peer. The TOE does not accept certificates that cannot be determined to be valid.
- PKGTLS11:FCS_TLSC_EXT.2: The TOE authenticates its TLS peer using x509v3 certificates and presents a certificate at the request of its peer.
- PKGTLS11:FCS_TLSC_EXT.5: The TOE supports only curves secp256r1 and secp384r1 for use in TLS communications.

6.3 User data protection

The TOE enforces a media transmission policy between registered VVoIP endpoints. The VVoIP endpoints are identified by an E.164 telephone number (SIP alias). The TOE mediates the creation of SRTP channels between registered VVoIP endpoints, ensuring that both endpoints are properly identified. The TOE permits SRTP traffic only when the following conditions are true:

- the client TOE must be registered with the ESC,
- a call has been established with a VVoIP endpoint,
- the TOE is not in the 'off-hook' state, and
- the TOE is not in the 'mute' state.

The TOE allows messages to be sent securely between two VVoIP endpoints. The TOE uses the same SIP-TLS protections on the secure text messages sent between two users of the TOE application.

The TOE enforces no additional information flow control policy rules, nor does it explicitly authorize or deny any information flows.

All TCP and UDP ports previously used by the TOE are closed when a call is terminated.

The User data protection function satisfies the following security functional requirements:

- ASPP13:FDP_DAR_EXT.1: The TOE protects sensitive data by storing secret and private keys using platform provided secure key storage.
- ASPP13:FDP_DEC_EXT.1: The TOE restricts its access the mobile device network, camera, microphone, Bluetooth, proximity sensor and notifications. Bluetooth does not have its own permission. The proximity sensor also does not have its own permission, and it is used only to disable touch and turn off the backlight. The TOE also restricts its access to the address book.
- VVoIPASEP10:FDP_IFC.1: The TOE enforces its voice/video media transmission policy as described above.
- VVoIPASEP10:FDP_IFF.1: The TOE enforces its voice/video media transmission policy as described above.
- ASPP13:FDP_NET_EXT.1: The TOE restricts its network communication to include only registration of the client with an SCA, initiation of an outgoing call, accepting/rejecting an incoming call, and sending/receiving a secure text message.

6.4 Identification and authentication

The Identification and authentication function satisfies the following security functional requirements:

- ASPP13:FIA_X509_EXT.1: Certificates are validated as part of the authentication process when they are presented to the TOE and when they are loaded into the TOE. The TOE requires the certificate to contain a SAN extension. The following fields are verified as appropriate: SAN checks, key usages, chain validation, expiration status, and revocation status. Chain validation includes ensuring that all certificates representing a CA indicate so using the basicConstraints CA flag having a value of TRUE. Revocation checking is also performed using a Certificate Revocation List. Wildcards are not allowed in certificates.
- ASPP13:FIA_X509_EXT.2: The TOE utilizes certificates automatically, which are also provisioned by the ESC SIP Server during registration. The TOE uses certificates for mutual authentication with the ESC. Certificates are checked and if found not valid, the certificates are not accepted. If an otherwise valid certificate cannot have its CRL checked to confirm that the certificate is not revoked, the certificate is accepted.

6.5 Security management

The SIP client and server use username and passwords to allow the client to access the SIP server and SCA server (separate passwords). These passwords are transmitted during initial SCA registration in the SIP Settings response and are stored persistently by the TOE. The TOE uses soft key storage to protect these secrets.

During initial start-up, the user needs to enter an activation code to start the initial registration. The Activation code is a shared secret between the server and the client used only during initial registration. Without the initial registration the client application can not communicate with the SIP server. The client does not have any default credentials.

The key chain of the TOE soft key store anchors in a secret stored to the platform key stores (key chain) and hence the user must unlock the keystore (using platform featues such as fingerprint or device password) to open the application.

The TOE supports the following management functions[1]:

- Specify the SIP Server to use for connections. User can enter SCA Server address during the activation phase which determines the SIP server. Note: This is the only parameter that the user can configure via the TOE.

The SCA server pushes configuration settings including username and password during registration. This configuration is created on the ESC and is used by the TOE. These setting configure the following:

- SIP Password which is a randomly generated 24 character alphanumeric values that is generated on the SCA server and transmistted within the SIP settings response of the SCA protocol.
- Configure cryptographic algorithms associated with protocols mandated in this PP. The algorithms are selected by the SIP server during TLS handshake.
- Load X5.09v3 certificates used for security functions in this PP – the client's X.509v3 certificates are generated during registration with the SCA server.
- Configure certificate revocation check. There are no configurable parameters for revocation checking.

The client device platform performs the following management functions:

- Ability to update the TOE, and to verify the updates. The user may uninstall or update the TOE using the mobile device operating system and app store. Downloaded TOE updates (apps) are verified using digital signatures in accordance with supported mobile device Security Targets for Android and iOS platforms.

The Security management function satisfies the following security functional requirements:

- ASPP13:FMT_CFG_EXT.1: Prior to registration with the SCA server, the client is presented with the activation screen and cannot leave that screen until a valid Activation Code is presented to the SIP server.
- ASPP13:FMT_MEC_EXT.1: The TOE stores configuration options for Android platforms in an XML file at location /data/data/package/shared_prefs,. For iOS platforms the TOE stores configuration options using the user defaults system or key-value store. TOE configuraiton is stored locally, but established and modified by the remote SCA or SIP server. Neither the TOE nor platform offer the ability to configure the TOE locally.
- ASPP13:FMT_SMF.1: The TOE provides management functions identified above.
- VVoIPASEP10:FMT_SMF.1: The TOE provides the following management operations:
    1) Specify the SIP Server to use for connections.

6.6 Privacy

The Privacy function satisfies the following security functional requirements:

- ASPP13:FPR_ANO_EXT.1: The TOE does not collect any PII and does not intentionally transmit any PII over a network.

6.7 Protection of the TSF

The TOE is physically protected by the boundary of the evaluated device. The TOE is executed on an evaluated Android 8.0/8.1 or iOS 12 device. Under Android, the TOE is constructed as an Java Application that executes most security relevant code in the native layer implemented in C/C++ (as a shared library). This Java code primary purpose is to provide the UI Implementation. For iOS, the TOE is constructed as a native application implemented in objective C and C++.

Memory mapping and permissions on memory regions are not functions applicable to a Java application. However, some 3rd party libraries are written in a language other than Java and thus are subject to the requirement for Anti-Exploitation Capabilities. However, none of the 3rd party libraries used by the TOE request memory mapping at explicit addresses, and none allocate memory for both write and execute permission.

Android's application management requires application updates to be signed with an Android key, thus allowing the secure updates of its applications. The Android OS Linux kernel is capable of ASLR (address space layout randomization), ensuring that no application uses the same address layout on two different devices.

The TOE libraries are also compiled with the '-fstack-protector-all -fno-exceptions' flags in order to enable ASLR and stack-based buffer over flow protections. On iOS the ASLR feature (-pie) is not set by a compiler flag, because it is on by default on the C-language compiler and this setting is required by the Apple App store.

The TOE is assigned a version[2] number by the vendor which is constructed using the following convention.

<major release>.<minor release>. <build number>.<year><week><debug>

For example, a release number might be 3.0.244.19100. This is interpreted as follows:

Major: 3
Minor: 0
Build: 244
Year: 2019
Week:10
Debug:0 (0:release; 1:debug build)

Major numbers represent significant product changes, while minor numbers are incremented for feature improvements and bug fixes. The Build number, year, week and debug state round out the identification of the software providing increased uniqueness.

The product includes the following 3rd party libraries:

- Android & iOS
  - webrtc 1.0.0
  - Boost 1.67.0
  - ZLIB 1.2.11
  - BZip2 1.0.8
  - Openssl 1.0.2r
  - pjproject 2.7.1
  - PocoCpp 1.9.1
  - libphonenumber 7.7.3
  - opus 1.1.3
  - silk 1.0.9
  - libsrtp 1.5.4

Android Only
- joda-time 2.1
- spongycastle 1.54.0.0
- androidannotations 4.6.0
- firebase 20.0.0
- androidx.appcompat 1.1.0
- zxing barcode scanner 3.6.0
- kotlin-stdlib-jdk7 1.3.50
- androidx.biometric 1.0.0-beta01

The user may install or update the TOE using the mobile device operating system and app store. Alternatively, an MDM may be used to push the TOE app and updates to the user's mobile device – in such cases user interaction/acceptance is still required. TOE updates are signed by with the BlackBerry software signing key associated with each platform. Downloaded TOE updates (apps) are verified using digital signatures in accordance with supported mobile device Security Targets.

The Protection of the TSF function satisfies the following security functional requirements:

- ASPP13:FPT_AEX_EXT.1: The TOE does not make requests to map memory at an explicit address, nor does it allocate any memory region with both write and execute permissions. Refer to the above discussion for more information.
- ASPP13:FPT_API_EXT.1: The TOE uses platform services by using only documented platform provided APIs. The specific interfaces used by the TOE on Android and iOS platforms are provided in section 7.

[2] The vendor refers to this identifier as a release number.

- ASPP13:FPT_IDV_EXT.1: The TOE identifies software as described above.
- ASPP13:FPT_LIB_EXT.1: The TOE includes the 3rd party libraries listed above.
- ASPP13:FPT_TUD_EXT.1/ASPP13:ALC_TSU_EXT.1: The TOE relies upon the platform to provide a mechanism that can check for product updates, to query the current version of the TOE, and to support the installation of an update as described above. The Android platform uses the Google Play Store, and the iOS platform uses the Apple App Store. Both app stores require developers to digitally sign their applications with a key recognized by each respective app store. This signature is used to verify that the application is from a trusted source. BlackBerry provides timely security updates for the TOE in case vulnerabilities have been discovered. Reported vulnerabilities and defects are investigated and rated based on the threat and result of the impact analysis and then scheduled for an upcoming bug fix release based on the severity. BlackBerry aims for a security update between 10 and a maximum of 50 days. Third party library updates are also included as a part of the TOE's update. BlackBerry accepts vulnerability reports through the BlackBerry form at https://www.blackberry.com/us/en/forms/enterprise/contact-us.
- ASPP13:FPT_TUD_EXT.2: The TOE is distributed using the Android application package (APK) format on an Android platform and using the IPA format on an iOS platform. These platforms force the TOE to write all data within the application working directory (sandbox), thus ensuring the application's removal results in the deletion of all traces of the application, with the exception of configuration settings, output files, and audit/log events

6.8 TOE access

The ESC enforces a default timeout of 15 seconds for the SRTP channel. The ESC allow configuration of the SRTP timeout value.

The TOE access function satisfies the following security functional requirements:

- VVoIPASEP10:FTA_SSL.3(1): The configuration loaded into the TOE during registration includes the configured interval for termination of an idle a voice/video transmission.

6.9 Trusted path/channels

The Trusted path/channels function satisfies the following security functional requirements:

- ASPP13:FTP_DIT_EXT.1: This SFR is superceded by FTP_DIT_EXT.1 from VVoIPASEP10, which allows communication using SRTP as well as TLS.
- VVoIPASEP10:FTP_DIT_EXT.1: The TOE encrypts all data transmitted between itself, an Enterprise Session Controller and another VVoIP endpoint using SIP-TLS and SRTP. The TOE uses these protocols to protect SIP signaling, VoIP calls, and secure text messaging.
- VVoIPASEP10:FTP_ITC.1/Control: The TOE uses SIP for all data transmitted between itself and an Enterprise Session Controller for the purpose of communicating with another VVoIP endpoint. The SIP communication occurs within the context of an already established TLS session. The TOE or ESC can initiate SIP communication in the existing TLS session.
- VVoIPASEP10:FTP_ITC.1/Media: The TOE uses SRTP as a trusted channel to communicate with a proxy on an ESC. Using this proxy, the TOE can communicate with another VVoIP endpoint for media traffic. The STRP cryptographic parameters are passed between the TOE and the other VVoIP endpoint using SIP messages protected by TLS (traffic that is part of FTP_ITC.1/Control). Using the proxy provided by the ESC, either the TOE or the other VVoIP endpoint can initiate transmission of voice/video media traffic.

7. API used by the TOE

The TOE uses platform services by using only documented platform provided APIs.

7.1 Android platform interfaces invoked by the TOE

The following are the platform APIs invoked by the TOE when running on an Android platform.

android.animation.Animator
android.animation.AnimatorListenerAdapter
android.animation.ArgbEvaluator
android.animation.LayoutTransition
android.animation.ObjectAnimator
android.animation.PropertyValuesHolder
android.animation.ValueAnimator
android.annotation.SuppressLint
android.annotation.TargetApi
android.app.Activity
android.app.ActivityManager
android.app.AlarmManager
android.app.AlertDialog
android.app.Application
android.app.Dialog
android.app.IntentService
android.app.KeyguardManager
android.app.Notification
android.app.NotificationChannel
android.app.NotificationManager
android.app.PendingIntent
android.app.Service
android.app.TimePickerDialog
android.bluetooth.BluetoothAdapter
android.bluetooth.BluetoothHeadset
android.bluetooth.BluetoothProfile
android.content.ActivityNotFoundException
android.content.BroadcastReceiver
android.content.ClipboardManager
android.content.ClipData
android.content.ComponentName
android.content.ContentResolver
android.content.ContentValues
android.content.Context
android.content.CursorLoader
android.content.DialogInterface
android.content.Intent
android.content.IntentFilter
android.content.pm.ActivityInfo
android.content.pm.PackageInfo
android.content.pm.PackageManager
android.content.res.ColorStateList
android.content.res.Configuration
android.content.res.Resources
android.content.RestrictionsManager
android.content.res.TypedArray
android.content.res.XmlResourceParser
android.content.ServiceConnection
android.content.SharedPreferences
android.database.ContentObserver
android.database.Cursor
android.database.DataSetObserver
android.graphics.Bitmap
android.graphics.BitmapFactory
android.graphics.Canvas
android.graphics.Color
android.graphics.ColorFilter
android.graphics.DashPathEffect
android.graphics.drawable.BitmapDrawable
android.graphics.drawable.Drawable
android.graphics.LinearGradient
android.graphics.Matrix
android.graphics.Paint
android.graphics.PathEffect
android.graphics.PixelFormat
android.graphics.PorterDuff
android.graphics.PorterDuffXfermode
android.graphics.Rect
android.graphics.RectF
android.graphics.Region
android.graphics.Shader
android.graphics.Typeface
android.hardware.fingerprint.FingerprintManager
android.hardware.Sensor
android.hardware.SensorEvent
android.hardware.SensorEventListener
android.hardware.SensorManager
android.Manifest
android.media.AudioDeviceInfo
android.media.AudioManager
android.media.AudioRecordingConfiguration
android.media.ExifInterface
android.media.MediaPlayer
android.media.RingtoneManager
android.media.ThumbnailUtils
android.net.ConnectivityManager
android.net.NetworkInfo
android.net.Uri
android.net.wifi.WifiManager
android.os.Binder

android.os.Build
android.os.Bundle
android.os.Environment
android.os.Handler
android.os.IBinder
android.os.Looper
android.os.Parcelable
android.os.PowerManager
android.os.PowerManager.WakeLock
android.os.SystemClock
android.os.Vibrator
android.preference.PreferenceManager
android.provider.CallLog
android.provider.ContactsContract
android.provider.ContactsContract
android.provider.DocumentsContract
android.provider.MediaStore
android.provider.Settings
android.provider.Telephony
android.security.KeyPairGeneratorSpec
android.security.keystore.KeyGenParameterSpec
android.security.keystore.KeyPermanentlyInvalidatedException
android.security.keystore.KeyProperties
android.security.keystore.KeyProtection
android.telephony.PhoneNumberFormattingTextWatcher
android.telephony.PhoneNumberUtils
android.telephony.SmsManager
android.telephony.SmsMessage
android.telephony.TelephonyManager
android.text.Editable
android.text.format.DateFormat
android.text.Html
android.text.Layout
android.text.Spannable
android.text.SpannableString
android.text.style.AlignmentSpan
android.text.style.StyleSpan
android.text.TextUtils
android.text.TextWatcher
android.util.AttributeSet
android.util.Base64
android.util.Base64
android.util.Log
android.util.Log
android.util.LongSparseArray
android.util.Pair
android.util.SparseArray
android.util.SparseBooleanArray
android.util.TypedValue
android.view.ActionMode
android.view.animation.AccelerateInterpolator
android.view.animation.Animation
android.view.animation.AnimationUtils
android.view.animation.DecelerateInterpolator
android.view.animation.ScaleAnimation
android.view.ContextMenu
android.view.Display
android.view.GestureDetector
android.view.Gravity
android.view.inputmethod.InputMethodManager
android.view.KeyEvent
android.view.LayoutInflater
android.view.Menu
android.view.MenuInflater
android.view.MenuItem
android.view.MotionEvent
android.view.SubMenu
android.view.View
android.view.ViewGroup
android.view.ViewGroup.LayoutParams
android.view.View.OnClickListener
android.view.View.OnFocusChangeListener
android.view.View.OnLongClickListener
android.view.View.OnTouchListener
android.view.Window
android.view.WindowManager
android.widget.AbsListView
android.widget.Adapter
android.widget.AdapterView
android.widget.AdapterView.OnItemClickListener
android.widget.AdapterView.OnItemLongClickListener
android.widget.ArrayAdapter
android.widget.BaseAdapter
android.widget.BaseExpandableListAdapter
android.widget.Button
android.widget.Checkable
android.widget.CheckBox
android.widget.CheckedTextView
android.widget.Chronometer
android.widget.EditText
android.widget.ExpandableListView
android.widget.Filter
android.widget.Filterable
android.widget.FrameLayout
android.widget.ImageButton
android.widget.ImageView
android.widget.LinearLayout
android.widget.ListAdapter
android.widget.ListView
android.widget.ProgressBar
android.widget.RelativeLayout
android.widget.ScrollView
android.widget.SectionIndexer

android.widget.SeekBar
android.widget.SeekBar.OnSeekBarChangeListener
android.widget.TextView
android.widget.TimePicker
android.widget.Toast
androidx.annotation.CallSuper
androidx.annotation.NonNull
androidx.annotation.Nullable
androidx.appcompat.app.ActionBar
androidx.appcompat.app.ActionBarDrawerToggle
androidx.appcompat.app.AlertDialog
androidx.appcompat.app.AppCompatActivity
androidx.appcompat.graphics.drawable.DrawerArrowDrawable
androidx.appcompat.widget.SearchView
androidx.biometric.BiometricConstants
androidx.biometric.BiometricManager
androidx.biometric.BiometricPrompt
androidx.biometric.BiometricPrompt.AuthenticationCallback
androidx.core.app.ActivityCompat
androidx.core.app.NotificationCompat
androidx.core.content.ContextCompat
androidx.core.view.GestureDetectorCompat
androidx.core.view.MenuItemCompat
androidx.core.view.ViewCompat
androidx.core.view.ViewPropertyAnimatorListener
androidx.drawerlayout.widget.DrawerLayout
androidx.fragment.app.DialogFragment
androidx.fragment.app.Fragment
androidx.fragment.app.FragmentActivity
androidx.fragment.app.FragmentManager
androidx.fragment.app.ListFragment
androidx.lifecycle.Lifecycle
androidx.lifecycle.LifecycleObserver
androidx.lifecycle.OnLifecycleEvent
androidx.lifecycle.ProcessLifecycleOwner
androidx.localbroadcastmanager.content.LocalBroadcastManager
androidx.swiperefreshlayout.widget.SwipeRefreshLayout
java.io.BufferedReader
java.io.ByteArrayInputStream
java.io.ByteArrayOutputStream
java.io.File
java.io.FileInputStream
java.io.FileOutputStream
java.io.FileReader
java.io.InputStream
java.io.InputStreamReader
java.io.IOException
java.io.IOException
java.io.Serializable
java.io.Serializable
java.io.UnsupportedEncodingException
java.lang.reflect.Field
java.lang.reflect.Method
java.math.BigInteger
java.net.Inet4Address
java.net.Inet6Address
java.net.InetAddress
java.net.NetworkInterface
java.nio.ByteBuffer
java.nio.file.Path
java.nio.file.Paths
java.security.cert.Certificate
java.security.cert.CertificateException
java.security.cert.CertificateFactory
java.security.cert.CertificateParsingException
java.security.cert.X509Certificate
java.security.InvalidAlgorithmParameterException
java.security.KeyFactory
java.security.KeyPair
java.security.KeyPairGenerator
java.security.KeyStore
java.security.KeyStoreException
java.security.NoSuchAlgorithmException
java.security.NoSuchProviderException
java.security.Principal
java.security.PrivateKey
java.security.PublicKey
java.security.SecureRandom
java.security.Security
java.security.Signature
java.security.spec.PKCS8EncodedKeySpec
java.security.spec.X509EncodedKeySpec
java.text.DateFormat
java.text.ParseException
java.text.SimpleDateFormat
java.util.ArrayList
java.util.Arrays
java.util.Calendar
java.util.Collection
java.util.Collections
java.util.Comparator
java.util.concurrent.atomic.AtomicBoolean
java.util.concurrent.atomic.AtomicInteger
java.util.concurrent.Callable
java.util.concurrent.ConcurrentHashMap
java.util.concurrent.ConcurrentSkipListMap
java.util.concurrent.CopyOnWriteArrayList
java.util.concurrent.CopyOnWriteArraySet
java.util.concurrent.Executors
java.util.concurrent.ExecutorService
java.util.concurrent.Future
java.util.concurrent.LinkedBlockingQueue

java.util.concurrent.TimeUnit
java.util.Date
java.util.Enumeration
java.util.HashMap
java.util.HashSet
java.util.Iterator
java.util.List
java.util.Locale
java.util.Map
java.util.Queue
java.util.regex.Matcher
java.util.regex.Pattern
java.util.Scanner
java.util.Set
java.util.SortedMap
java.util.StringTokenizer
java.util.UUID
java.util.zip.ZipEntry
java.util.zip.ZipInputStream
javax.crypto.BadPaddingException
javax.crypto.Cipher
javax.crypto.CipherInputStream
javax.crypto.CipherOutputStream
javax.crypto.IllegalBlockSizeException
javax.crypto.KeyGenerator
javax.crypto.SecretKey
javax.crypto.spec.IvParameterSpec
javax.security.auth.x500.X500Principal
javax.xml.parsers.SAXParser
javax.xml.parsers.SAXParserFactory

7.2 iOS platform interfaces invoked by the TOE

The following are the paltform interfaces invoked by the TOE when running on an iOS platform.

Payload/SecuSUITE.app/SecuSUITE:

/System/Library/Frameworks/PushKit.framework/PushKit (compatibility version 1.0.0, current version 1.0.0)
/System/Library/Frameworks/AddressBook.framework/AddressBook (compatibility version 1.0.0, current version 1.0.0)
/System/Library/Frameworks/AddressBookUI.framework/AddressBookUI (compatibility version 1.0.0, current version 33.0.0)
/System/Library/Frameworks/AudioToolbox.framework/AudioToolbox (compatibility version 1.0.0, current version 492.0.0)
/System/Library/Frameworks/AVFoundation.framework/AVFoundation (compatibility version 1.0.0, current version 2.0.0)
/System/Library/Frameworks/CFNetwork.framework/CFNetwork (compatibility version 1.0.0, current version 975.0.3)
/System/Library/Frameworks/Contacts.framework/Contacts (compatibility version 0.0.0, current version 0.0.0)
/System/Library/Frameworks/CoreAudio.framework/CoreAudio (compatibility version 1.0.0, current version 1.0.0)
/System/Library/Frameworks/CoreGraphics.framework/CoreGraphics (compatibility version 64.0.0, current version 1245.9.2)
/System/Library/Frameworks/CoreTelephony.framework/CoreTelephony (compatibility version 1.0.0, current version 0.0.0)
/System/Library/Frameworks/Foundation.framework/Foundation (compatibility version 300.0.0, current version 1560.10.0)
/System/Library/Frameworks/UIKit.framework/UIKit (compatibility version 1.0.0, current version 61000.0.0)
/System/Library/Frameworks/SystemConfiguration.framework/SystemConfiguration (compatibility version 1.0.0, current version 963.200.27)
/System/Library/Frameworks/Accelerate.framework/Accelerate (compatibility version 1.0.0, current version 4.0.0)
/System/Library/Frameworks/CoreFoundation.framework/CoreFoundation (compatibility version 150.0.0, current version 1560.10.0)
/System/Library/Frameworks/QuartzCore.framework/QuartzCore (compatibility version 1.2.0, current version 1.11.0)
/System/Library/Frameworks/Security.framework/Security (compatibility version 1.0.0, current version 58286.222.2)

/System/Library/Frameworks/UserNotifications.framework/UserNotifications (compatibility version 1.0.0, current version 1.0.0)
/usr/lib/libc++.1.dylib (compatibility version 1.0.0, current version 400.9.4)
/usr/lib/libobjc.A.dylib (compatibility version 1.0.0, current version 228.0.0)