Protection Profile for Voice Over IP (VoIP) Applications

21 October 2013

Version 1.2

Table of Contents

List of Tables

List of Figures

Revision History

Version	Date	Description
0.6	*January 2013*	Initial release
1.2	*October 2013*	TOE refinement to include an API that enables the VoIP Application to interact with the device platform. Structured selected requirements and assurance activities to reflect whether they need to be satisfied by the TOE, the operational environment (the platform), or either.

1 INTRODUCTION

This Protection Profile (PP) supports procurements of commercial off-the-shelf (COTS) VoIP Client Applications to provide secure tunnels to authenticated remote endpoints or servers. This PP details the policies, assumptions, threats, security objectives, security functional requirements, and security assurance requirements for the VoIP Application and its supporting environment.

The primary intent is to clearly communicate to developers the Security Functional Requirements needed to counter the threats that are being addressed by the VoIP Application. The description in the TOE Summary Specification (TSS) of the Security Target (ST) is expected to document the architecture of the product (Target of Evaluation) and the mechanisms used to ensure that critical security transactions are correctly implemented.

1.1 Overview of the TOE

This document specifies Security Functional Requirements (SFRs) for a VoIP Application. VoIP provides a protected transmission of private voice data between two endpoints. The VoIP application in the context of this PP is part of a workspace installed for use by the phone user. The VoIP infrastructure can vary greatly, both in size and complexity. Many kinds of functionality are possible, often desirable, and sometimes necessary – including Session Border Controllers (SBC), gateways, trunking, and Network Address Translation (NAT) and firewall traversal. The VoIP Application in the context of this PP is considered to be a VoIP client that interacts with a SIP Server which provides registrar and proxy capabilities required for call-session management via SIP requests and responses to establish, process, and terminate VoIP calls.

The TOE defined by this PP is the VoIP Client Application, a component executing on a remote access client, together with an API that enables the VoIP client application to interact with other applications and the client device platform (part of the Operational Environment of the TOE).

1.2 Usage of the TOE

The VoIP Application is intended to provide a secure tunnel to a remote VoIP Application. The tunnel provides confidentiality, integrity, and data authentication for information that travels across the public network. The VoIP Application will interact with a peer VoIP Application using the Security 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. All VoIP Applications that comply with this document will support SDES-SRTP. Likewise, compliant TOEs must also protect 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 is required by this PP to make use of certificates to authenticate both the SIP server end and the TOE itself through the TLS connection.

As shown in Figure 1, the TOE communicates with other VoIP clients and SIP Servers over protected channels. Components in red are addressed in this PP. Components in blue are addressed in related PPs.

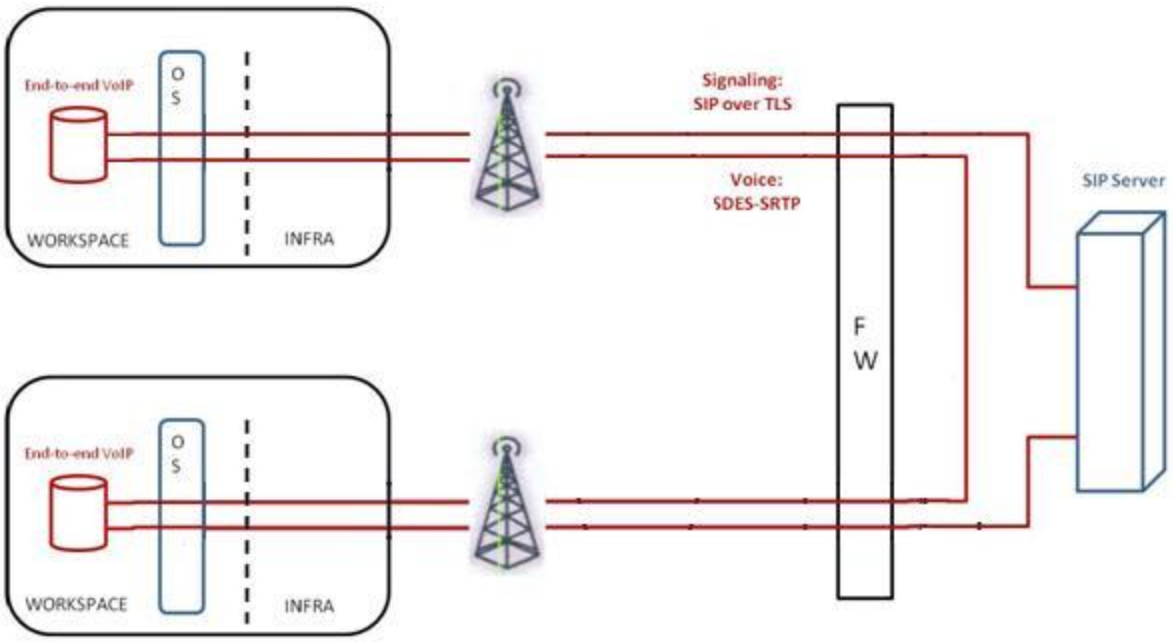

Figure 1: VoIP Communication

It is assumed that the VoIP Application is implemented properly and contains no critical design mistakes. The VoIP Application relies on the operational environment for its proper execution as well as the following client machine protection mechanisms: audit review, audit storage, identification and authentication, security management, and session management. The vendor is required to provide configuration guidance (AGD_PRE, AGD_OPE) to correctly install and administer the client machine and the TOE for every operational environment supported.

2 SECURITY PROBLEM DESCRIPTION

This PP is written to address the situation in which a user relies on a public network to establish secure voice communication. To protect the data in-transit from disclosure and modification, a VoIP tunnel is created to establish secure communications. The VoIP Application provides one end of the secure VoIP tunnel and performs encryption and decryption of network packets. The proper installation, configuration, and updating of the VoIP Application are critical to its correct operation, so these objectives are included as well.

ANNEX A: SUPPORTING TABLES presents the Security Problem Description (SPD) in a more “traditional” form.

2.1 Unauthorized Access to User and TOE Data (T.UNAUTHORIZED_ACCESS)

This PP 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 client device on which the VoIP Application is installed. Therefore, the primary threat agents are the unauthorized entities that try to gain access.

The remote endpoint of the voice communication can be both geographically and logically distant from the TOE, and 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 and/or manipulated directly by intermediate systems, leading to a compromise of the TOE. SDES-SRTP can be used to provide protection for this communication; however, there are a myriad of 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 compliant with the protocol specification but will not be able to interact with other, diverse equipment that is typically found in large enterprises.

Even though the communication path is protected, there is a possibility that the remote SIP Server could be duped into thinking that 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 SIP Server as if it were the TOE. In a similar manner, the TOE could also be duped into thinking that it is establishing communications with a legitimate remote SIP Server 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 attacker 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.

2.2 Inability to Configure the TSF (T.TSF_CONFIGURATION)

Configuring VoIP 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 to configure certain aspects of the interface may also lead to the mis-specification of the desired communications policy or use of cryptography that may be desired or required for a particular users’ site. This may result in unintended weak or plain-text 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 VoIP Application.

2.3 Malicious Updates (T.UNAUTHORIZED_UPDATE)

Since one of the most common attack vectors used involves attacking unpatched versions of software containing well-known flaws, updating the VoIP client applicaiton is necessary to ensure that changes to threat environment are addressed. Timely application of patches ensures that the client is a “hard target”, thus increasing the likelihood that product will be able to maintain and enforce its security policy. However, the updates to be applied to the product must be trustable in some manner; otherwise, an attacker can write their own “update” that instead contains malicious code of their choosing, such as a rootkit, bot, or other malware. Once this “update” is installed, the attacker then has control of the system and all of its data.

Methods of countering this threat typically involve hashes of the updates, and potentially cryptographic operations (e.g., digital signatures) on those hashes as well. However, the validity of these methods introduces additional threats. For instance, a weak hash function could result in the attacker being able

to modify the legitimate update in such a way that the hash remained unchanged. For cryptographic signature schemes, there are dependencies on

1) the strength of the cryptographic algorithm used to provide the signature, and
2) the ability of the end user to verify the signature (which typically involves checking a hierarchy of digital signatures back to a root of trust (a certificate authority)).

If a cryptographic signature scheme is weak, then it may be compromised by an attacker and the end user will install a malicious update, thinking that it is legitimate. Similarly, if the root of trust can be compromised, then a strong digital signature algorithm will not stop the malicious update from being installed (the attacker will just create their own signature on the update using the compromised root of trust, and the malicious update will then be installed without detection).

2.4 TSF Failure (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 SECURITY OBJECTIVES

Compliant TOEs will provide security functionality that address threats to the TOE and implements policies that are imposed by law or regulation. The following sections provide a description of this functionality in light of the threats previously discussed that motivate its inclusion in compliant TOEs. The Security Objectives are the requirements for the Target of Evaluation (TOE) and for the Operational Environment derived from Section 2.

3.1 Establish Secure Tunnels

To address the issues concerning transmitting sensitive data between the TOE and the SIP Server or remote VoIP Application described in Section 2.1, compliant TOEs will provide an encrypted channel for these communication paths between themselves and the SIP Server or remote VoIP Application. These channels are implemented using TLS and SDES-SRTP. TLS and SDES-SRTP are specified by RFCs that offer a variety of implementation choices. Requirements have been imposed on some of these choices (particularly those for cryptographic primitives) to provide interoperability and resistance to cryptographic attack. While compliant TOEs must support all of the choices specified in the ST, they may support additional algorithms and protocols. If such additional mechanisms are not evaluated, guidance must be given to the administrator to make clear the fact that they are not evaluated.

In addition to providing protection from disclosure (and detection of modification) for the communications, TLS and SDES-SRTP offers two-way authentication of each endpoint in a cryptographically secure manner, meaning that even if there was a malicious attacker between the two endpoints, any attempt to represent themselves to either endpoint of the communications path as the other communicating party would be detected. This authentication is done using X.509 certificates to provide greater assurance in the authentication. The requirements on the TLS and SDES-SRTP protocols, in addition to the structure of the protocols themselves, provide protection against replay attacks such as those described in Section 2.1.

(O.SECURE_TUNNEL → FCS_CKM.1(1), FCS_CKM.1(2), FCS_CKM_EXT.4, FCS_COP.1(1), FCS_COP.1(2), FCS_COP.1(3), FCS_COP.1(4), FIA_SIPC_EXT.1, FCS_SRTP_EXT.1, FCS_TLS_EXT.1, FIA_X509_EXT.1, FTP_ITC.1)

3.2 Configuration of the TOE

To address the issues concerning the configuration of the TOE described in Section 2.2, the TOE will provide interfaces to control the configuration of TLS and SDES-SRTP and the underlying cryptographic mechanisms supporting the protocol, management of X.509 certificates, and updates to the TOE.

(O.TOE_CONFIGURATION → FMT_SMF.1)

3.3 Verifiable Updates

As outlined in Section 2.3, failure to verify that updates to the client can be trusted may lead to compromise of the security functionality. A first step in establishing trust in the update is to publish a hash of the update that can be verified prior to installing the update. While this establishes that the update downloaded is the one associated with the published hash, it does not indicate if the source of the update/hash combination has been compromised or can't be trusted. To establish trust in the source of the updates, a cryptographic mechanisms and procedures to procure the update, check the update cryptographically through the TOE-provided digital signature mechanism, and install the update will be provided.

(O.VERIFIABLE_UPDATES → FPT_TUD_EXT.1, FCS_COP.1(2), FCS_COP.1(3), FIA_X509_EXT.1)

3.4 TSF Self Test

In order to detect some number of failures of underlying security mechanisms used by the TSF, the TSF will perform self-tests. The extent of this self testing is left to the product developer, but a more comprehensive set of self tests should result in a more trustworthy platform on which to develop enterprise architecture.

(O.TSF_SELF_TEST → FPT_TST_EXT.1)

4 SECURITY REQUIREMENTS

The Security Functional Requirements included in this section are derived from Part 2 of the *Common Criteria for Information Technology Security Evaluation, Version 3.1, Revision 4* (the CC), with additional extended functional components. The Security Assurance Requirements included in this section are derived from Part 3 of the CC. Supplemental Guidance is provided in the form of Assurance Activities associated with the functional requirements in Sections 4.2 and 4.3, as well as with the Security Assurance Requirements themselves in Section 4.4.

4.1 Conventions

The CC defines operations on Security Functional Requirements (SFR): assignments, selections, assignments within selections and refinements. This document uses the following font conventions to identify the operations defined by the CC:

- Assignment: Indicated with *italicized* text;
- Refinement made by PP author: Indicated by the word "Refinement" in **bold text** after the element number with additional **bold** text and deletions with strikethroughs, if necessary;
- Selection: Indicated with underlined text;
- Assignment within a Selection: Indicated with *italicized and underlined* text;
- Iteration: Indicated by appending the iteration number in parenthesis, e.g., (1), (2), (3).

Explicitly stated SFRs are identified by having a label 'EXT' after the requirement name for TOE SFRs.

4.2 Security Functional Requirements for VoIP Applications (TOE)

Security functional requirements in the main body of this PP are divided into those that must be satisfied by the VoIP Application (the TOE), and those that must be satisfied by either the TOE or the platform on which it runs. This section contains the requirements that must be met by the TOE.

4.2.1 Cryptographic Support (FCS)

FCS_CKM.2(1) Refinement: Cryptographic Key Storage

FCS_CKM_EXT.2(1) Cryptographic Key Storage

FCS_CKM_EXT.2.1(1) The VoIP client application shall store persistent secrets and private keys when not in use in platform-provided key storage.

Application Note:
This requirement ensures that persistent secrets and private keys are stored securely when not in use.

This requirement assumes persistent secrets and private keys used by the VoIP client application will be stored by the platform.

Assurance Activity:

The evaluator shall examine the TSS to ensure it describes in detail how user credentials, certificates, persistent secret and private keys are stored. The evaluator reviews the TSS to determine that it makes a case that key material is not written unencrypted to persistent memory, and that key material is stored by the platform.

FCS_SRTP_EXT.1 Secure Real-Time Transport Protocol (SRTP)

FCS_SRTP_EXT.1.1 The VoIP client application 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.

FCS_SRTP_EXT.1.2 The VoIP client application shall implement SDES-SRTP supporting the following ciphersuites in accordance with RFC 4568: AES_CM_128_HMAC_SHA1_80.

FCS_SRTP_EXT.1.3 The VoIP client application shall ensure the SRTP NULL algorithm can be disabled.

FCS_SRTP_EXT.1.4 The VoIP client application shall allow the SRTP ports to be used for SRTP communications to be specified by an Authorized Administrator.

Application Note:

This requirement specifies that the SRTP session that will be used to carry the VoIP traffic will be keyed according to an SDES dialog using the identified ciphersuite. In the future Suite B ciphersuites will be available.

Assurance Activity:

The evaluator shall examine the TSS to verify that it describes how the SRTP session is negotiated for both incoming and outgoing calls. This includes how the keying material is established, as well has how requests to use the NULL algorithm or other unallowed ciphersuites are rejected by the TSF. The evaluator shall also perform the following test:

- *Test 1: The evaluator shall follow the procedure for initializing their device so that they are ready to receive and place calls. The evaluator shall then both place and receive a call and determine that the traffic sent and received by the TOE is encrypted. To ensure that the call is being encrypted and to view the ciphersuites being used a packet capture tool should be used. In order to decrypt the TLS-SIP traffic and view the SDES negotiation the SIP server's private key needs to be loaded into the packet capture tool.*

4.2.2 User Data Protection (FDP)

FDP_VOP_EXT.1 Voice Over IP Data Protection

FDP_VOP_EXT.1.1 The VoIP Client Application shall stop the transmission of voice data when a VoIP call is placed on hold, a VoIP call is placed on mute, a VoIP call is not connected, and [assignment: other actions, no other actions].

***Assurance Activity**:*

The evaluator shall examine the TSS to verify that it describes how each of the functions in the requirement is implemented. The evaluator shall also perform the following tests:

- *Test 1: The evaluator shall follow the procedure for initializing the device so that it is ready to receive and place calls. Using a packet capture tool, the evaluator shall verify that no voice traffic is transmitted until a call is placed/received. The evaluator shall place a call and verify that the voice traffic is being sent through the secure channel. The evaluator shall then implement each*

of the functions listed (mute, hold, disconnect, and any other specified actions) and verify that voice traffic is no longer being transmitted.

- *Test 2: The evaluator shall follow the procedure for initializing the device so that it is ready to receive and place calls. Using a packet capture tool, the evaluator shall verify that no voice traffic is transmitted until a call is placed/received. The evaluator shall receive a call and verify that the voice traffic is being sent through the secure channel. The evaluator shall then implement each of the functions listed (mute, hold, disconnect, and any other specified actions) and verify that voice traffic is no longer being transmitted.*

4.2.3 Identification and Authentication (FIA)

The baseline requirements for the TOE are fairly limited with respect to I&A, since no formal administrative or general purpose users are defined. The extent of the I&A required to be performed by the TOE relates to the authentication done at the machine level when establishing the TLS, and SDES/SRTP connections. Therefore, the requirements in this section cover only the credentials that are used by the protocols specified in this PP.

FIA_SIPC_EXT.1 Session Initiation Protocol (SIP) Client

FIA_SIPC_EXT.1.1 The VoIP client application shall implement the Session Initiation Protocol (SIP) that complies with RFC 3261 using the Session Description Protocol (SDP) complying with RFC 4566 to describe the multimedia session that will be used to carry the VOIP traffic.

FIA_SIPC_EXT.1.2 The VoIP client application shall require the user to enter a password to support the use of password authentication for SIP REGISTER function requests as specified in section 22 of RFC 3261.

FIA_SIPC_EXT.1.3 The VoIP client application shall support SIP authentication passwords that contain at least [assignment: positive integer of 8 or more] characters in the set of {upper case characters, lower case characters, numbers, and the following special characters: “!”, “@”, “#”, “$”, “%”, “^”, “&”, “*”, “(“, and “)”, and [assignment: other supported special characters]}.

FIA_SIPC_EXT.1.4 The password entered by the user as per FIA_SIPC_EXT.1.2 shall be cleared by the VoIP client application once the VoIP client application is notified that the REGISTER request was successful.

Application Note:

The only SIP request that is required to be authenticated (by the SIP Server) is the REGISTER request; the TOE supports this by providing a user-entered password. While the SIP Server will perform the enforcement and only register the user upon the presentation of the correct password, the client is required by the elements above to support passwords that are at least 8 characters long (the maximum length is defined in the first assignment) and can contain the characters identified in FIA_SIPC_EXT.1.3 (characters allowed by the TOE but not listed explicitly in the element should be identified in the second assignment; otherwise “no other characters” is an acceptable assignment), and to prompt the user for the password when it sends the REGISTER request.

The intent of the FIA_SIPC_EXT.1.4 element is that the plaintext password used for SIP registration is not maintained on the device. It is acceptable to store values derived from this password (e.g., a hash) that can be used if an additional REGISTER function needs to be sent.

Assurance Activity:

The evaluator shall examine the TSS to verify that it describes how the SIP session is established. This shall include the initiation of the SIP session, registration of the user, and how both outgoing and incoming calls are handled (initiated, described, and terminated). This description shall also include a description of the handling of the password from the time it is entered by the user until the time it is cleared by the TSF.

The evaluator shall also perform the following tests:

- *Test 1: The evaluator shall follow the procedure for initializing their device to include establishing a connection to the SIP Server. The evaluator shall confirm that they are prompted for a password prior to successfully completing the SIP REGISTER request.*
- *Test 2: The evaluator shall follow the procedure for initializing their device to include establishing a connection to the SIP Server. The evaluator shall confirm that entering an incorrect password results in the device not being registered by the SIP Server (e.g., they are unable to successful place or receive calls). The evaluator shall also confirm that entering the correct password allows the successful registration of the device (e.g., by being able to place and receive calls).*
- *Test 3: The evaluator shall set up the test environment such that a variety of passwords are shown to be accepted by the TOE, such that the length and character set identified in FIA_SIPC_EXT.1.3 is represented. The test report shall contain a rationale by the evaluator that the test set used is representative of the allowed lengths and characters.*

4.2.4 Security Management (FMT)

The TOE is not required to maintain a separate management role. It is, however, required to provide functionality to configure certain aspects of TOE operation that should not be available to the general user population.

FMT_SMF.1 Specification of Management Functions

FMT_SMF.1.1 The VoIP client application shall be capable of performing the following management functions:

- Specify the SIP Server to use for connections,
- Specify VoIP client credentials to be used for connections,
- Specify password requirements for SIP authentication,
- Ability to configure all security management functions identified in other sections of this PP,
- [selection: visual alert notification configuration, [assignment: any additional management functions], no other functions].

Application Note:
For installation, the VoIP client application relies on the IT environment to authenticate the administrator to the client machine.

Assurance Activity:

The evaluator shall check to make sure that every management function mandated by the PP is described in the operational guidance and that the description contains the information required to perform the management duties associated with the management function. The evaluator shall test the TOE's ability to provide the management functions by configuring the TOE and testing each option listed in the requirement above. The evaluator is expected to test this functions in all the ways in which the ST and guidance documentation state the configuration can be managed.

Note that the testing here may be accomplished in conjunction with the testing of other requirements, such as FCS_TLS_EXT.1.

4.2.5 Protection of the TSF (FPT)

FPT_TUD_EXT.1 Extended: Trusted Update

FPT_TUD_EXT.1.1 The TSF shall provide the client device platform the ability to query the current version of the TOE firmware/software.

Application Note:
Any update of the TOE will be handled by a function of the platform and is not a function of the TOE itself. However, the TOE must have the ability to correctly report its version to the platform in order to facilitate decisions on whether to perform the update.

Assurance Activity:
The evaluator shall check the TSS to determine that it describes the method by which the TOE reports its current version. The TOE guidance shall contain the invocation sequence necessary to obtain the current version of the TOE.

The evaluator shall perform the following tests:

- *Test 1: The evaluator shall invoke the platform functionality to query the current version of the TOE. The evaluator shall confirm that the current version of the TOE is returned.*

4.2.6 Trusted Path/Channel (FTP)

FTP_ITC.1(1) Inter-TSF Trusted Channel (SDES-SRTP)

FTP_ITC.1.1(1) **Refinement:** The VoIP Client Application shall provide a communication channel between itself and a **remote VoIP application using SDES-SRTP as specified in FCS_SRTP_EXT.1** that is logically distinct from other communication channels and provides assured identification of its end points and protection of the channel data from modification **and** disclosure.

FTP_ITC.1.2(1) The VoIP Client Application shall permit the TSF or the remote VoIP application to initiate communication via the trusted channel.

FTP_ITC.1.3(1) The VoIP Client Application shall initiate communication via the trusted channel for [all communications between the two devices].

Application Note:

This requirement addresses the case where the communications is established between a VoIP Application on another device and the TOE, both of which act as peers in the protocol sense.

The requirement implies that not only are communications protected when they are initially established, but also on resumption after an outage. It may be the case that some part of the TOE setup involves manually setting up tunnels to protect other communications, and if after an outage the TOE attempts to re-establish the communication automatically with (the necessary) manual intervention, there may be a window created where an attacker might be able to gain critical information or compromise a connection.

***Assurance Activity**:*

The evaluator shall examine the TSS section to confirm that it describes how this requirement is implemented in the TOE, along with TOE-specific options or procedures that might not be reflected in the specification. The evaluator shall also confirm that all protocols listed in the TSS are specified and included in the requirements in the ST. The evaluator shall confirm that the operational guidance contains instructions for establishing the connection to the peer, and that it contains recovery instructions should a connection be unintentionally broken. The evaluator shall verify that communication can be initiated from both the TSF and the remote VoIP Application. The evaluator shall also perform the following tests:

- *Test 1: The evaluators shall ensure that the TOE is able to initiate communications with a remote VoIP Application using the protocols specified in the requirement, setting up the connections as described in the operational guidance and ensuring that communication is successful.*
- *Test 2: The evaluator shall ensure, for each communication channel with a remote VoIP Application, the channel data is not sent in plaintext.*
- *Test 3: The evaluator shall ensure, for each communication channel with a remote VoIP Application, modification of the channel data is detected by the TOE.*
- *Test 4: The evaluators shall physically interrupt the connection from the TOE to the remote VoIP Application. The evaluators shall ensure that subsequent communications are appropriately protected, at a minimum in the case of any attempts to automatically resume the connection or connect to a new remote VoIP Application.*

Further assurance activities are associated with the specific protocols.

4.3 Security Functional Requirements for VoIP Client Applications or Client Platforms

Security functional requirements in the main body of this PP are divided into those that must be satisfied by the VoIP Application (the TOE), and those that must be satisfied by either the TOE or the platform on which it runs. This section contains requirements that must be met, but they can either be met by the TOE or the platform on which the TOE operates. Each requirement includes a selection for the ST author to indicate whether the VoIP Client Application or the client platform performs the functionality in the requirement. In the case where the TOE relies on the platform, the platform must be evaluated either concurrently with or before the VoIP Client Application. Assurance activities are therefore separated into those that apply when the requirements are met by the TOE, and those that are performed when the platform on which the TOE operates implements the required functionality. If

a test or documentation assurance activity is specified that is not specifically associated with either the TOE or the TOE platform, then it applies regardless of where the requirement is implemented.

It should be noted that several protocols are used during call establishment: TLS, SIP, SDP, and SDES-SRTP. While these protocols (and associated TSS and Testing Assurance Activities) are specified separately, it is expected that a comprehensive description and end-to-end test case/cases can be used to describe and demonstrate these capabilities.

4.3.1 Cryptographic Support (FCS)

FCS_CKM.1(1) Cryptographic Key Generation (Asymmetric Keys)

FCS_CKM.1.1(1) **Refinement:** The [selection, choose at least one of: VoIP client application, client device platform] shall generate **asymmetric** cryptographic keys **used for key establishment** in accordance with:

- NIST Special Publication 800-56B, "Recommendation for Pair-Wise Key Establishment Schemes Using Integer Factorization Cryptography" for RSA-based key establishment schemes and

[selection:

- NIST Special Publication 800-56A, "Recommendation for Pair-Wise Key Establishment Schemes Using Discrete Logarithm Cryptography" for finite field-based key establishment schemes;
- NIST Special Publication 800-56A, "Recommendation for Pair-Wise Key Establishment Schemes Using Discrete Logarithm Cryptography" for elliptic curve-based key establishment schemes and implementing "NIST curves" P-256, P-384 and [selection: P-521, no other curves] (as defined in FIPS PUB 186-4, "Digital Signature Standard");
- No other algorithms]

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

Application Note:
This component requires that the TOE/platform be able to generate the public/private key pairs that are used for key establishment purposes for the various cryptographic protocols used.

Since the domain parameters to be used are specified by the requirements of the protocol in this PP, it is not expected that the TOE/platform will generate domain parameters, and therefore there is no additional domain parameter validation needed when complying with the protocols specified in this PP.

Assurance Activity:

Requirement met by the platform
For each platform listed in the ST, the evaluator shall examine the ST of the platform to ensure that the key establishment claimed in that platform's ST contains the key establishment requirement in the VoIP Client Application's ST. The evaluator shall also examine the TSS of the VoIP Client Application's ST to verify that it describes (for each supported platform) how the key establishment functionality is invoked (it should be noted that this may be through a mechanism that is not implemented by the VoIP Application; nonetheless, that mechanism will be identified in the TSS as part of this assurance activity).

Requirement met by the TOE

This assurance activity will verify the key generation and key establishments schemes used on the TOE.

Key Generation:

The evaluator shall verify the implementation of the key generation routines of the supported schemes using the applicable tests below.

Key Generation for RSA-Based Key Establishment Schemes

The evaluator shall verify the implementation of RSA Key Generation by the TOE using the Key Generation test. This test verifies the ability of the TSF to correctly produce values for the key components including the public verification exponent e, the private prime factors p and q, the public modulus n and the calculation of the private signature exponent d.

Key Pair generation specifies 5 ways (or methods) to generate the primes p and q. These include:

1. *Random Primes:*
   - *Provable primes*
   - *Probable primes*
2. *Primes with Conditions:*
   - *Primes p1, p2, q1,q2, p and q shall all be provable primes*
   - *Primes p1, p2, q1, and q2 shall be provable primes and p and q shall be probable primes*
   - *Primes p1, p2, q1,q2, p and q shall all be probable primes*

To test the key generation method for the Random Provable primes method and for all the Primes with Conditions methods, the evaluator must seed the TSF key generation routine with sufficient data to deterministically generate the RSA key pair. This includes the random seed(s), the public exponent of the RSA key, and the desired key length. For each key length supported, the evaluator shall have the TSF generate 25 key pairs. The evaluator shall verify the correctness of the TSF's implementation by comparing values generated by the TSF with those generated from a known good implementation.

Key Generation for Finite-Field Cryptography (FFC) – Based 56A Schemes

FFC Domain Parameter and Key Generation Tests

The evaluator shall verify the implementation of the Parameters Generation and the Key Generation for FFC by the TOE using the Parameter Generation and Key Generation test. This test verifies the ability of the TSF to correctly produce values for the field prime p, the cryptographic prime q (dividing p-1), the cryptographic group generator g, and the calculation of the private key x and public key y.

The Parameter generation specifies 2 ways (or methods) to generate the cryptographic prime q and the field prime p:

Cryptographic and Field Primes:

- *Primes q and p shall both be provable primes*
- *Primes q and field prime p shall both be probable primes*

and two ways to generate the cryptographic group generator g:

Cryptographic Group Generator:

- *Generator g constructed through a verifiable process*
- *Generator g constructed through an unverifiable process.*

The Key generation specifies 2 ways to generate the private key x:

Private Key:

- *len(q) bit output of RBG where 1 <=x <= q-1*

- *len(q) + 64 bit output of RBG, followed by a mod q-1 operation where 1<= x<=q-1.*

The security strength of the RBG must be at least that of the security offered by the FFC parameter set.

To test the cryptographic and field prime generation method for the provable primes method and/or the group generator g for a verifiable process, the evaluator must seed the TSF parameter generation routine with sufficient data to deterministically generate the parameter set.

For each key length supported, the evaluator shall have the TSF generate 25 parameter sets and key pairs. The evaluator shall verify the correctness of the TSF's implementation by comparing values generated by the TSF with those generated from a known good implementation. Verification must also confirm

- *g != 0,1*
- *q divides p-1*
- *g^q mod p = 1*
- *g^x mod p = y*

for each FFC parameter set and key pair.

Key Generation for Elliptic Curve Cryptography (ECC) - Based 56A Schemes

ECC Key Generation Test

For each supported NIST curve, i.e., P-256, P-284 and P-521, the evaluator shall require the implementation under test (IUT) to generate 10 private/public key pairs. The private key shall be generated using an approved random bit generator (RBG). To determine correctness, the evaluator shall submit the generated key pairs to the public key verification (PKV) function of a known good implementation.

ECC Public Key Verification (PKV) Test

For each supported NIST curve, i.e., P-256, P-284 and P-521, the evaluator shall generate 10 private/public key pairs using the key generation function of a known good implementation and modify five of the public key values so that they are incorrect, leaving five values unchanged (i.e., correct). The evaluator shall obtain in response a set of 10 PASS/FAIL values.

Key Establishment Schemes

The evaluator shall verify the implementation of the key establishment schemes of the supported by the TOE using the applicable tests below.

SP800-56A Key Establishment Schemes

The evaluator shall verify a TOE's implementation of SP800-56A key agreement schemes using the following Function and Validity tests. These validation tests for each key agreement scheme verify that a TOE has implemented the components of the key agreement scheme according to the specifications in the Recommendation. These components include the calculation of the DLC primitives (the shared secret value Z) and the calculation of the derived keying material (DKM) via the Key Derivation Function (KDF). If key confirmation is supported, the evaluator shall also verify that the components of key confirmation have been implemented correctly, using the test procedures described below. This includes the parsing of the DKM, the generation of MACdata and the calculation of MACtag.

Function Test

The Function test verifies the ability of the TOE to implement the key agreement schemes correctly. To conduct this test the evaluator shall generate or obtain test vectors from a known good implementation of the TOE supported schemes. For each supported key agreement scheme-key agreement role combination, KDF type, and, if supported, key confirmation role- key confirmation type combination, the tester shall generate 10 sets of test vectors. The data set consists of one set of domain parameter values (FFC) or the NIST approved curve (ECC) per 10 sets of public keys. These keys are static, ephemeral or both depending on the scheme being tested.

The evaluator shall obtain the DKM, the corresponding TOE's public keys (static and/or ephemeral), the MAC tag(s), and any inputs used in the KDF, such as the Other Information field OI and TOE id fields.

If the TOE does not use a KDF defined in SP 800-56A, the evaluator shall obtain only the public keys and the hashed value of the shared secret.

The evaluator shall verify the correctness of the TSF's implementation of a given scheme by using a known good implementation to calculate the shared secret value, derive the keying material DKM, and compare hashes or MAC tags generated from these values.

If key confirmation is supported, the TSF shall perform the above for each implemented approved MAC algorithm.

Validity Test

The Validity test verifies the ability of the TOE to recognize another party's valid and invalid key agreement results with or without key confirmation. To conduct this test, the evaluator shall obtain a list of the supporting cryptographic functions included in the SP800-56A key agreement implementation to determine which errors the TOE should be able to recognize. The evaluator generates a set of 24 (FFC) or 30 (ECC) test vectors consisting of data sets including domain parameter values or NIST approved curves, the evaluator's public keys, the TOE's public/private key pairs, MACTag, and any inputs used in the KDF, such as the other info and TOE id fields.

The evaluator shall inject an error in some of the test vectors to test that the TOE recognizes invalid key agreement results caused by the following fields being incorrect: the shared secret value Z, the DKM, the other information field OI, the data to be MACed, or the generated MACTag. If the TOE contains the full or partial (only ECC) public key validation, the evaluator will also individually inject errors in both parties' static public keys, both parties' ephemeral public keys and the TOE's static private key to assure the TOE detects errors in the public key validation function and/or the partial key validation function (in ECC only). At least two of the test vectors shall remain unmodified and therefore should result in valid key agreement results (they should pass).

The TOE shall use these modified test vectors to emulate the key agreement scheme using the corresponding parameters. The evaluator shall compare the TOE's results with the results using a known good implementation verifying that the TOE detects these errors.

SP800-56B Key Establishment Schemes

At this time, detailed test procedures for RSA-based key establishment schemes are not available. In order to show that the TSF complies with 800-56A and/or 800-56B, depending on the selections made, the evaluator shall ensure that the TSS contains the following information:

- *The TSS shall list all sections of the appropriate 800-56 standard(s) to which the TOE complies.*
- *For each applicable section listed in the TSS, for all statements that are not "shall" (that is, "shall*

not", "should", and "should not"), if the TOE implements such options it shall be described in the TSS. If the included functionality is indicated as "shall not" or "should not" in the standard, the TSS shall provide a rationale for why this will not adversely affect the security policy implemented by the TOE.

For each applicable section of 800-56A and 800-56B (as selected), any omission of functionality related to "shall" or "should" statements shall be described.

FCS_CKM.1(2) Cryptographic Key Generation

FCS_CKM.1.1(2) **Refinement:** The [selection, choose at least one of: VoIP client application, client device platform] shall generate **asymmetric** cryptographic keys **used for authentication** in accordance with a specified cryptographic key generation algorithm [selection:

- FIPS PUB 186-4, "Digital Signature Standard (DSS)", Appendix B.3 for RSA schemes;
- FIPS PUB 186-4, "Digital Signature Standard (DSS)", Appendix B.4 for ECDSA schemes and implementing "NIST curves" P-256, P-384 and [selection: P-521, no other curves];
- ANSI X9.31-1998, Appendix A.2.4 Using AES for RSA schemes]

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

Application Note:
While it is expected that the public key generated be associated with an identity in an X509v3 certificate, this association is not required to be performed by the TOE, and instead is expected to be performed by a Certificate Authority in the Operational Environment.
For elliptic curve-based schemes, the key size refers to the log_2 of the order of the base point.

The ANSI X9.31-1998 option will be removed from the selection in a future publication of this document. Presently, the selection is not exclusively limited to the FIPS PUB 186-4 options in order to allow industry some further time to complete the transition to the modern FIPS PUB 186-4 standard. As the preferred approach for cryptographic signature, elliptic curves will be required in future publications of this PP.

See NIST Special Publication 800-57, "Recommendation for Key Management" for information about equivalent key strengths.

Assurance Activity:

Requirement met by the platform

For each platform listed in the ST, the evaluator shall examine the ST of the platform to ensure that the key generation function claimed in that platform's ST contains the key generation requirement in the VoIP Client Application's ST. The evaluator shall also examine the TSS of the VoIP Client Application's ST to verify that it describes (for each supported platform) how the key generation functionality is invoked (it should be noted that this may be through a mechanism that is not implemented by the VoIP Application; nonetheless, that mechanism will be identified in the TSS as part of this assurance activity).

Requirement met by the TOE

If the TSF implements a FIPS 186-4 signature scheme, this requirement is verified under FCS_COP.1(2).

If the ESF implements the ANSI X9.31-1998 scheme, the evaluator shall check to ensure that the TSS describes how the key-pairs are generated. In order to show that the TSF implementation complies with ANSI X9.31-1998, the evaluator shall ensure that the TSS contains the following information:

- *The TSS shall list all sections of the standard to which the TOE complies;*
- *For each applicable section listed in the TSS, for all statements that are not "shall" (that is, "shall not", "should", and "should not"), if the TOE implements such options it shall be described in the TSS. If the included functionality is indicated as "shall not" or "should not" in the standard, the TSS shall provide a rationale for why this will not adversely affect the security policy implemented by the TOE;*

For each applicable section of Appendix B, any omission of functionality related to "shall" or "should" statements shall be described.

FCS_CKM_EXT.4 Cryptographic key material destruction (Key Material)

FCS_CKM_EXT.4.1 **Refinement:** The [selection, choose at least one of: VoIP client application, client device platform] shall zeroize all plaintext secret and private cryptographic keys and Critical Security Parameters (CSPs) when no longer required.

Application Note:
Any security related information (such as keys, authentication data, and passwords) must be zeroized when no longer in use to prevent the disclosure or modification of security critical data.

"Critical Security Parameters" are defined in FIPS 140-2 as "security-related information (e.g., secret and private cryptographic keys, and authentication data such as passwords and PINs) whose disclosure or modification can compromise the security of a cryptographic module."

The zeroization indicated above applies to each intermediate storage area for plaintext key/cryptographic critical security parameter (i.e., any storage, such as memory buffers, that is included in the path of such data) upon the transfer of the key/cryptographic critical security parameter to another location.

In practice, the TOE will not implement all of the functionality associated with the requirement, since if it performs zeroization at all it will be by invoking platform interfaces to perform the storage location clear/overwrite function. The ST author should select "TOE" when, for at least one of the keys needed to meet the requirements of this PP, the TOE manipulates (reads, writes) the data identified in the requirement and thus needs to ensure that those data are cleared. In these cases, it is sufficient for the TOE to invoke the correct underlying functions of the host to perform the zeroization--it does not imply that the TOE has to include a kernel-mode memory driver to ensure the data are zeroized.

In the likely event that some of the data are manipulated by the TOE and other data are manipulated entirely by the platform, the ST author shall iterate this requirement.

Assurance Activity:

Requirement met by the platform

The evaluator shall check to ensure the TSS describes each of the secret keys (keys used for symmetric encryption), private keys, and CSPs used to generate key that are not otherwise covered by the FCS_CKM_EXT.4 requirement levied on the TOE.

For each platform listed in the ST, the evaluator shall examine the TSS of the ST of the platform to ensure that each of the secret keys, private keys, and CSPs used to generate key listed above are covered.

Requirement met by the TOE

The evaluator shall check to ensure the TSS describes each of the secret keys (keys used for symmetric encryption), private keys, and CSPs used to generate key; when they are zeroized (for example, immediately after use, on system shutdown, etc.); and the type of zeroization procedure that is performed (overwrite with zeros, overwrite three times with random pattern, etc.). If different types of memory are used to store the materials to be protected, the evaluator shall check to ensure that the TSS describes the zeroization procedure in terms of the memory in which the data are stored (for example, "secret keys stored on flash are zeroized by overwriting once with zeros, while secret keys stored on the internal hard drive are zeroized by overwriting three times with a random pattern that is changed before each write"). If a read-back is done to verify the zeroization, this shall be described as well.

For each key clearing situation described in the TSS the evaluator shall repeat the following test.

- *Test 1: The evaluator shall utilize appropriate combinations of specialized operational environment and development tools (debuggers, simulators, etc.) for the TOE and instrumented TOE builds to test that keys are cleared correctly, including all intermediate copies of the key that may have been created internally by the TOE during normal cryptographic processing with that key.*

  Cryptographic TOE implementations in software shall be loaded and exercised under a debugger to perform such tests. The evaluator shall perform the following test for each key subject to clearing, including intermediate copies of keys that are persisted encrypted by the TOE:

  - *Load the instrumented TOE build in a debugger.*
  - *Record the value of the key in the TOE subject to clearing.*
  - *Cause the TOE to perform a normal cryptographic processing with the key from #1.*
  - *Cause the TOE to clear the key.*
  - *Cause the TOE to stop the execution but not exit.*
  - *Cause the TOE to dump the entire memory footprint of the TOE into a binary file.*
  - *Search the content of the binary file created in #4 for instances of the known key value from #1.*

  The test succeeds if no copies of the key from #1 are found in step #7 above and fails otherwise.

  The evaluator shall perform this test on all keys, including those persisted in encrypted form, to ensure intermediate copies are cleared.

- *Test 2: In cases where the TOE is implemented in firmware and operates in a limited operating environment that does not allow the use of debuggers, the evaluator shall utilize a simulator for the TOE on a general purpose operating system. The evaluator shall provide a rationale explaining the instrumentation of the simulated test environment and justifying the obtained test results.*

FCS_COP.1(1) Cryptographic Operation (Data Encryption/Decryption)

FCS_COP.1.1 **Refinement:** The [selection, choose at least one of: VoIP client application, client device platform] shall perform *encryption and decryption* in accordance with a specified cryptographic algorithm *AES operating in* **CTR, CBC,** *and [selection: GCM (as defined in NIST SP800-38D), [assignment: one or more modes], no other modes]* and cryptographic key sizes 128-bits, and *[selection: 256-bits, 192 bits, no other key sizes]* that meets the following:

- *FIPS PUB 197, "Advanced Encryption Standard (AES)"*
- *NIST SP 800-38A, [selection: NIST SP 800-38D, no others]*

Application Note:
This PP requires CTR and CBC modes to be used in the SDES and TLS protocols (FCS_SRTP, FCS_TLS). Therefore, the FCS_COP.1.1(1) element has been specified here to ensure the ST Author includes these two modes to be consistent with the protocol requirements in the PP.

For the assignment, the ST author should choose the mode or modes in which AES operates. For the first selection, the ST author should choose the key sizes that are supported by this functionality. For the second selection, the ST author should choose the standards that describe the modes specified in the assignment.

Assurance Activity:

Requirement met by the platform
For each platform listed in the ST, the evaluator shall examine the ST of the platform to ensure that the encryption/decryption function(s) claimed in that platform's ST contains the encryption/decryption function(s) in the VoIP Client Application's ST. The evaluator shall also examine the TSS of the VoIP Client Application's ST to verify that it describes (for each supported platform) how the encryption/decryption functionality is invoked for each mode and key size selected in the VoIP Client Application's ST (it should be noted that this may be through a mechanism that is not implemented by the VoIP Client Application; nonetheless, that mechanism will be identified in the TSS as part of this assurance activity).

Requirement met by the TOE
The evaluator shall perform the following activities based on the selections in the ST.

AES-CTR Tests

The evaluator shall use tests appropriate to CTR modes from "The Advanced Encryption Standard Algorithm Validation Suite (AESAVS)" (these documents are available from http://csrc.nist.gov/groups/STM/cavp/index.html) as a guide in testing the requirement above. This will require that the evaluator have a trusted reference implementation of the algorithms that can produce test vectors that are verifiable during the test.

AES-CBC Tests

AES-CBC Known Answer Tests

There are four Known Answer Tests (KATs), described below. In all KATs, the plaintext, ciphertext, and IV values shall be 128-bit blocks. The results from each test may either be obtained by the evaluator directly or by supplying the inputs to the implementer and receiving the results in response. To determine correctness, the evaluator shall compare the resulting values to those obtained by submitting the same inputs to a known good implementation.

> ***KAT-1.*** *To test the encrypt functionality of AES-CBC, the evaluator shall supply a set of 10 plaintext values and obtain the ciphertext value that results from AES-CBC encryption of the given plaintext using a key value of all zeros and an IV of all zeros. Five plaintext values shall be encrypted with a 128-bit all-zeros key, and the other five shall be encrypted with a 256-bit all-zeros key.*

To test the decrypt functionality of AES-CBC, the evaluator shall perform the same test as for encrypt, using 10 ciphertext values as input and AES-CBC decryption.

KAT-2. *To test the encrypt functionality of AES-CBC, the evaluator shall supply a set of 10 key values and obtain the ciphertext value that results from AES-CBC encryption of an all-zeros plaintext using the given key value and an IV of all zeros. Five of the keys shall be 128-bit keys, and the other five shall be 256-bit keys.*

To test the decrypt functionality of AES-CBC, the evaluator shall perform the same test as for encrypt, using an all-zero ciphertext value as input and AES-CBC decryption.

KAT-3. *To test the encrypt functionality of AES-CBC, the evaluator shall supply the two sets of key values described below and obtain the ciphertext value that results from AES encryption of an all-zeros plaintext using the given key value and an IV of all zeros. The first set of keys shall have 128 128-bit keys, and the second set shall have 256 256-bit keys. Key i in each set shall have the leftmost i bits be ones and the rightmost N-i bits be zeros, for i in [1,N].*

To test the decrypt functionality of AES-CBC, the evaluator shall supply the two sets of key and ciphertext value pairs described below and obtain the plaintext value that results from AES-CBC decryption of the given ciphertext using the given key and an IV of all zeros. The first set of key/ciphertext pairs shall have 128 128-bit key/ciphertext pairs, and the second set of key/ciphertext pairs shall have 256 256-bit key/ciphertext pairs. Key i in each set shall have the leftmost i bits be ones and the rightmost N-i bits be zeros, for i in [1,N]. The ciphertext value in each pair shall be the value that results in an all-zeros plaintext when decrypted with its corresponding key.

KAT-4. *To test the encrypt functionality of AES-CBC, the evaluator shall supply the set of 128 plaintext values described below and obtain the two ciphertext values that result from AES-CBC encryption of the given plaintext using a 128-bit key value of all zeros with an IV of all zeros and using a 256-bit key value of all zeros with an IV of all zeros, respectively. Plaintext value i in each set shall have the leftmost i bits be ones and the rightmost 128-i bits be zeros, for i in [1,128].*

To test the decrypt functionality of AES-CBC, the evaluator shall perform the same test as for encrypt, using ciphertext values of the same form as the plaintext in the encrypt test as input and AES-CBC decryption.

AES-CBC Multi-Block Message Test

The evaluator shall test the encrypt functionality by encrypting an i-block message where 1< i <=10. The evaluator shall choose a key, an IV and plaintext message of length i blocks and encrypt the message, using the mode to be tested, with the chosen key and IV. The ciphertext shall be compared to the result of encrypting the same plaintext message with the same key and IV using a known good implementation.

The evaluator shall also test the decrypt functionality for each mode by decrypting an i-block message where 1 < i <=10. The evaluator shall choose a key, an IV and a ciphertext message of length i blocks and decrypt the message, using the mode to be tested, with the chosen key and IV. The plaintext shall be compared to the result of decrypting the same ciphertext message with the same key and IV using a known good implementation.

AES-CBC Monte Carlo Tests

The evaluator shall test the encrypt functionality using a set of 200 plaintext, IV, and key 3-tuples. 100 of these shall use 128 bit keys, and 100 shall use 256 bit keys. The plaintext and IV values shall be 128-bit blocks. For each 3-tuple, 1000 iterations shall be run as follows:

```
# Input: PT, IV, Key
for i = 1 to 1000:
    if i == 1:
        CT[1] = AES-CBC-Encrypt(Key, IV, PT)
        PT = IV
    else:
        CT[i] = AES-CBC-Encrypt(Key, PT)
        PT = CT[i-1]
```

The ciphertext computed in the 1000th iteration (i.e., CT[1000]) is the result for that trial. This result shall be compared to the result of running 1000 iterations with the same values using a known good implementation.

The evaluator shall test the decrypt functionality using the same test as for encrypt, exchanging CT and PT and replacing AES-CBC-Encrypt with AES-CBC-Decrypt.

AES-GCM Monte Carlo Test

The evaluator shall test the authenticated encrypt functionality of AES-GCM for each combination of the following input parameter lengths:

128 bit and 256 bit keys

***Two plaintext lengths**. One of the plaintext lengths shall be a non-zero integer multiple of 128 bits, if supported. The other plaintext length shall not be an integer multiple of 128 bits, if supported.*

***Three AAD lengths**. One AAD length shall be 0, if supported. One AAD length shall be a non-zero integer multiple of 128 bits, if supported. One AAD length shall not be an integer multiple of 128 bits, if supported.*

***Two IV lengths**. If 96 bit IV is supported, 96 bits shall be one of the two IV lengths tested.*

The evaluator shall test the encrypt functionality using a set of 10 key, plaintext, AAD, and IV tuples for each combination of parameter lengths above and obtain the ciphertext value and tag that results from AES-GCM authenticated encrypt. Each supported tag length shall be tested at least once per set of 10. The IV value may be supplied by the evaluator or the implementation being tested, as long as it is known.

The evaluator shall test the decrypt functionality using a set of 10 key, ciphertext, tag, AAD, and IV 5-tuples for each combination of parameter lengths above and obtain a Pass/Fail result on authentication and the decrypted plaintext if Pass. The set shall include five tuples that Pass and five that Fail.

The results from each test may either be obtained by the evaluator directly or by supplying the inputs to the implementer and receiving the results in response. To determine correctness, the evaluator shall

compare the resulting values to those obtained by submitting the same inputs to a known good implementation.

FCS_COP.1(2) Cryptographic Operation (for cryptographic signature)

FCS_COP.1.1(2) **Refinement:** The [selection, choose at least one of: VoIP client application, client device platform] shall perform **cryptographic signature services (generation and verification)** in accordance with a specified cryptographic algorithm

- ***FIPS PUB 186-4, "Digital Signature Standard (DSS)", Appendix B.3 for RSA schemes***

[selection:

- ***FIPS PUB 186-4, "Digital Signature Standard (DSS)", Appendix B.4 for Elliptic Curve Digital Signature Algorithm (ECDSA) schemes and implementing "NIST curves" P-256, P-384, and [selection: P-521, no other curves]***
- ***No other algorithms]***

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

Application Note:

The ST Author should choose the algorithm implemented to perform digital signatures; if more than one algorithm is available, this requirement (and the corresponding FCS_CKM.1 requirement) should be iterated to specify the functionality. For the algorithm chosen, the ST author should make the appropriate assignments/selections to specify the parameters that are implemented for that algorithm.

RSA signature generation and verification is currently required in order to comply with FCS_TLS_EXT.1. As the preferred approach for digital signatures, ECDSA will be required in future publications of this PP.

Assurance Activity:

Requirement met by the platform

For each platform listed in the ST, the evaluator shall examine the ST of the platform to ensure that the digital signature functions claimed in that platform's ST contains the digital signature functions in the VoIP Client Application's ST. The evaluator shall also examine the TSS of the VoIP Client Application's ST to verify that it describes (for each supported platform) how the digital signature functionality is invoked for each operation they are used for in the VoIP client application (it should be noted that this may be through a mechanism that is not implemented by the VoIP Client Application; nonetheless, that mechanism will be identified in the TSS as part of this assurance activity).

Requirement met by the TOE

The evaluator shall perform the following activities based on the selections in the ST.

Key Generation:

Key Generation for RSA Signature Schemes

The evaluator shall verify the implementation of RSA Key Generation by the TOE using the Key Generation test. This test verifies the ability of the TSF to correctly produce values for the key components including the public verification exponent e, the private prime factors p and q, the public modulus n and the calculation of the private signature exponent d.

Key Pair generation specifies 5 ways (or methods) to generate the primes p and q. These include:

1. *Random Primes:*
   - *Provable primes*
   - *Probable primes*
2. *Primes with Conditions:*
   - *Primes p1, p2, q1,q2, p and q shall all be provable primes*
   - *Primes p1, p2, q1, and q2 shall be provable primes and p and q shall be probable primes*
   - *Primes p1, p2, q1,q2, p and q shall all be probable primes*

To test the key generation method for the Random Provable primes method and for all the Primes with Conditions methods, the evaluator must seed the TSF key generation routine with sufficient data to deterministically generate the RSA key pair. This includes the random seed(s), the public exponent of the RSA key, and the desired key length. For each key length supported, the evaluator shall have the TSF generate 25 key pairs. The evaluator shall verify the correctness of the TSF's implementation by comparing values generated by the TSF with those generated from a known good implementation.

ECDSA Key Generation Tests

FIPS 186-4 ECDSA Key Generation Test

For each supported NIST curve, i.e., P-256, P-284 and P-521, the evaluator shall require the implementation under test (IUT) to generate 10 private/public key pairs. The private key shall be generated using an approved random bit generator (RBG). To determine correctness, the evaluator shall submit the generated key pairs to the public key verification (PKV) function of a known good implementation.

FIPS 186-4 Public Key Verification (PKV) Test

For each supported NIST curve, i.e., P-256, P-284 and P-521, the evaluator shall generate 10 private/public key pairs using the key generation function of a known good implementation and modify five of the public key values so that they are incorrect, leaving five values unchanged (i.e., correct). The evaluator shall obtain in response a set of 10 PASS/FAIL values.

ECDSA Algorithm Tests

CDSA FIPS 186-4 Signature Generation Test

For each supported NIST curve (i.e., P-256, P-284 and P-521) and SHA function pair, the evaluator shall generate 10 1024-bit long messages and obtain for each message a public key and the resulting signature values R and S. To determine correctness, the evaluator shall use the signature verification function of a known good implementation.

ECDSA FIPS 186-4 Signature Verification Test

For each supported NIST curve (i.e., P-256, P-284 and P-521) and SHA function pair, the evaluator shall generate a set of 10 1024-bit message, public key and signature tuples and modify one of the values (message, public key or signature) in five of the 10 tuples. The evaluator shall obtain in response a set of 10 PASS/FAIL values.

RSA Signature Algorithm Tests

Signature Generation Test

The evaluator shall verify the implementation of RSA Signature Generation by the TOE using the Signature Generation Test. To conduct this test the evaluator must generate or obtain 10 messages from a trusted reference implementation for each modulus size/SHA combination supported by the TSF. The evaluator shall have the TOE use their private key and modulus value to sign these messages.

The evaluator shall verify the correctness of the TSF's signature using a known good implementation and the associated public keys to verify the signatures.

Signature Verification Test

The evaluator shall perform the Signature Verification test to verify the ability of the TOE to recognize another party's valid and invalid signatures. The evaluator shall inject errors into the test vectors produced during the Signature Verification Test by introducing errors in some of the public keys e, messages, IR format, and/or signatures. The TOE attempts to verify the signatures and returns success or failure.

The evaluator shall use these test vectors to emulate the signature verification test using the corresponding parameters and verify that the TOE detects these errors.

FCS_COP.1(3) Cryptographic Operation (for cryptographic hashing)

FCS_COP.1.1(3) **Refinement:** The [selection, choose at least one of: VoIP client application, client device platform] shall perform **cryptographic hashing** in accordance with a specified cryptographic algorithm SHA-1 and [selection: SHA-256, SHA-384, SHA-512, no other algorithms] and **message digest sizes** 160 bits and [selection: 256, 384, 512 bits, no other message digest sizes] that meet the following: *FIPS PUB 180-3, "Secure Hash Standard."*

Application Note:
In future versions of this document, SHA-1 may be removed as an option. SHA-1 for generating digital signatures will no longer be allowed after December 2013, and SHA-1 for verification of digital signatures is strong discouraged as there may be risk in accepting these signatures. SHA-1 is currently required in order to comply with FCS_TLS_EXT.1 and FCS_CKM.1.

The intent of this requirement is to specify the hashing function. The hash selection must support the message digest size selection. The hash selection should be consistent with the overall strength of the algorithm used (for example, SHA-256 for 128-bit keys).

Assurance Activity:

The evaluator checks the AGD documents to determine that any configuration that is required to be done to configure the functionality for the required hash sizes is present. The evaluator shall check that the association of the hash function with other TSF cryptographic functions (for example, the digital signature verification function) is documented in the TSS.

Requirement met by the platform

For each platform listed in the ST, the evaluator shall examine the ST of the platform to ensure that the hash function(s) claimed in that platform's ST contains the hash function(s) in the VoIP Client Application's ST. The evaluator shall also examine the TSS of the VoIP Client Application's ST to verify that it describes (for each supported platform) how the hash functionality is invoked for each digest size

selected in the VoIP Client Application's ST (it should be noted that this may be through a mechanism that is not implemented by the VoIP Client Application; nonetheless, that mechanism will be identified in the TSS as part of this assurance activity).

Requirement met by the TOE

The TSF hashing functions can be implemented in one of two modes. The first mode is the byte-oriented mode. In this mode the TSF only hashes messages that are an integral number of bytes in length; i.e., the length (in bits) of the message to be hashed is divisible by 8. The second mode is the bit-oriented mode. In this mode the TSF hashes messages of arbitrary length. As there are different tests for each mode, an indication is given in the following sections for the bit-oriented vs. the byte-oriented testmacs.

The evaluator shall perform all of the following tests for each hash algorithm implemented by the TSF and used to satisfy the requirements of this PP.

Short Messages Test - Bit-oriented Mode

The evaluators devise an input set consisting of m+1 messages, where m is the block length of the hash algorithm. The length of the messages range sequentially from 0 to m bits. The message text shall be pseudorandomly generated. The evaluators compute the message digest for each of the messages and ensure that the correct result is produced when the messages are provided to the TSF.

Short Messages Test - Byte-oriented Mode

The evaluators devise an input set consisting of m/8+1 messages, where m is the block length of the hash algorithm. The length of the messages range sequentially from 0 to m/8 bytes, with each message being an integral number of bytes. The message text shall be pseudorandomly generated. The evaluators compute the message digest for each of the messages and ensure that the correct result is produced when the messages are provided to the TSF.

Selected Long Messages Test - Bit-oriented Mode

*The evaluators devise an input set consisting of m messages, where m is the block length of the hash algorithm. The length of the ith message is 512 + 99*i, where 1 ≤ i ≤ m. The message text shall be pseudorandomly generated. The evaluators compute the message digest for each of the messages and ensure that the correct result is produced when the messages are provided to the TSF.*

Selected Long Messages Test - Byte-oriented Mode

*The evaluators devise an input set consisting of m/8 messages, where m is the block length of the hash algorithm. The length of the ith message is 512 + 8*99*i, where 1 ≤ i ≤ m/8. The message text shall be pseudorandomly generated. The evaluators compute the message digest for each of the messages and ensure that the correct result is produced when the messages are provided to the TSF.*

Pseudorandomly Generated Messages Test

This test is for byte-oriented implementations only. The evaluators randomly generate a seed that is n bits long, where n is the length of the message digest produced by the hash function to be tested. The evaluators then formulate a set of 100 messages and associated digests by following the algorithm provided in Figure 1 of [SHAVS]. The evaluators then ensure that the correct result is produced when the messages are provided to the TSF.

FCS_COP.1(4) Cryptographic Operation (For keyed-hash Message Authentication)

FCS_COP.1.1(4) **Refinement:** The [selection, choose at least one of: VoIP client application, client device platform] shall perform *keyed-hash message authentication* in accordance with a specified cryptographic algorithm *HMAC-SHA-1 and [selection:HMAC-SHA-256, HMAC- SHA-384,HMAC- SHA-512, no other algorithms]* and cryptographic key sizes *[assignment: key size (in bits) used in HMAC],* **and message digest sizes 160 and** [**selection: 256, 384, 512, no other**] **bits** that meet the following: *FIPS Pub 198-1, "The Keyed-Hash Message Authentication Code, and FIPS Pub 180-3, "Secure Hash Standard."*

Application note:
The selection in this requirement must be consistent with the key size specified for the size of the keys used in conjunction with the keyed-hash message authentication. HMAC-SHA-1 is currently required in order to comply with FCS_TLS_EXT.1 and FCS_CKM.1 but may be removed in future versions of this document.

The message digest size above corresponds to the underlying hash algorithm used. Note that truncating the output of the HMAC following the hash calculation is an appropriate step in a variety of applications. This does not invalidate compliance with this requirement, however, the ST should state that truncation is performed, the size of the final output, and the standard to which this truncation complies.

Assurance Activity:
The evaluator shall check that the association of the keyed-hash function with other cryptographic functions specified in the VoIP Client Application ST (whether these are performed by the platform or by the TOE) is documented in the TSS.

Requirement met by the platform
For each platform listed in the ST, the evaluator shall examine the ST of the platform to ensure that the keyed-hash function(s) claimed in that platform's ST contains the keyed-hash function(s) in the VoIP Client Application's ST. The evaluator shall also examine the TSS of the VoIP Client Application's ST to verify that it describes (for each supported platform) how the keyed-hash functionality is invoked for each mode and key size selected in the VoIP Client Application's ST (it should be noted that this may be through a mechanism that is not implemented by the VoIP Client Application; nonetheless, that mechanism will be identified in the TSS as part of this assurance activity).

Requirement met by the TOE
Additionally, for all cases where the output of the HMAC following the hash calculation is truncated, the evaluator shall ensure that the TSS states for what operation this truncation takes place; the size of the final output; and the standard to which this truncation complies.

The evaluator shall examine the TSS to ensure that it specifies the following values used by the HMAC function: key length, hash function used, block size, and output MAC length used.

For each of the supported parameter sets, the evaluator shall compose 15 sets of test data. Each set shall consist of a key and message data. The evaluator shall have the TSF generate HMAC tags for these sets of test data. The resulting MAC tags shall be compared to the result of generating HMAC tags with the same key and IV using a known good implementation.

FCS_RBG_EXT.1 Extended: Cryptographic operation (Random Bit Generation)

FCS_RBG_EXT.1.1 The [selection, choose at least one of: VoIP client application, client device platform] shall perform all deterministic random bit generation services in accordance with [selection, *choose one of: NIST Special Publication 800-90A using* [selection: *Hash_DRBG (any), HMAC_DRBG (any), CTR_DRBG (AES), Dual_EC_DRBG (any)]; FIPS Pub 140-2 Annex C: X9.31 Appendix 2.4 using AES*].

FCS_RBG_EXT.1.2 The deterministic RBG shall be seeded by an entropy source that accumulates entropy from [selection: *a software-based noise source, a platform-based RBG*] with a minimum of [selection: *128 bits, 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.

Application Note:

For the first selection in FCS_RBG_EXT.1.1 the ST author should select whether the TOE or the platform on which the TOE is installed provides the RBG services.

NIST Special Pub 800-90, Appendix C describes the minimum entropy measurement that will probably be required in future versions of FIPS-140. If possible this should be used immediately and be required in future versions of this PP.

For the second selection in FCS_RBG_EXT.1.1, the ST author should select the standard to which the RBG services comply (either 800-90 or 140-2 Annex C).

SP 800-90 contains four different methods of generating random numbers; each of these, in turn, depends on underlying cryptographic primitives (hash functions/ciphers). The ST author will select the function used (if 800-90 is selected), and include the specific underlying cryptographic primitives used in the requirement or in the TSS. While any of the identified hash functions (SHA-1, SHA-256, SHA-384, SHA-512) are allowed for Hash_DRBG or HMAC_DRBG, only AES-based implementations for CTR_DRBG are allowed. While any of the curves defined in 800-90 are allowed for Dual_EC_DRBG, the ST author not only must include the curve chosen, but also the hash algorithm used.

For the first selection in FCS_RBG_EXT.1.2, the ST author indicates whether the sources of entropy are software-based or platform-based, or both. If there are multiple sources of entropy, the ST will describe each entropy source and whether it is software or platform-based. Platform-based noise sources are preferred.

The platform-based RBG source is the output of a validated RBG provided by the platform, which is used as an entropy source for a TSF-provided DRBG according to FCS_RBG_EXT.1.1. In this way, the developer has chained RBGs as described in NIST SP800-90C.

Note that for FIPS Pub 140-2 Annex C, currently only the method described in NIST-Recommended Random Number Generator Based on ANSI X9.31 Appendix A.2.4 Using the 3-Key Triple DES and AES Algorithm, Section 3 is valid. If the key length for the AES implementation used here is different than that used to encrypt the user data, then FCS_COP.1 may have to be adjusted or iterated to reflect the different key length. For the selection in FCS_RBG_EXT.1.2, the ST author selects the minimum number of bits of entropy that is used to seed the RBG.

The ST author also ensures that any underlying functions are included in the baseline requirements for the TOE.

Assurance Activity:

Requirement met by the platform

For each platform listed in the ST, the evaluator shall examine the ST of the platform to ensure that the RBG functions claimed in that platform's ST contains the RBG functions in the VoIP Client Application's

ST. The evaluator shall also examine the TSS of the VoIP Client Application's ST to verify that it describes (for each supported platform) how the RBG functionality is invoked for each operation they are used for in the VoIP application (it should be noted that this may be through a mechanism that is not implemented by the VoIP application; nonetheless, that mechanism will be identified in the TSS as part of this assurance activity.

Requirement met by the TOE

Documentation shall be produced – and the evaluator shall perform the activities – in accordance with Annex E, Entropy and Documentation and Assessment.

If the ST author has selected a platform-based noise source, the evaluator shall verify that the platform's RBG has been validated by examining the platform's ST. The evaluator shall verify that the platform's RBG is seeded with at least the amount of entropy selected by the ST author for this profile. In this case, the ST author is not responsible for Annex E documentation of the platform's RBG.

The evaluator shall also perform the following tests, depending on the standard to which the RBG conforms.

Implementations Conforming to FIPS 140-2, Annex C

The reference for the tests contained in this section is The Random Number Generator Validation System (RNGVS). The evaluator shall conduct the following two tests. Note that the "expected values" are produced by a reference implementation of the algorithm that is known to be correct. Proof of correctness is left to each Scheme.

The evaluator shall perform a Variable Seed Test. The evaluator shall provide a set of 128 (Seed, DT) pairs to the TSF RBG function, each 128 bits. The evaluator shall also provide a key (of the length appropriate to the AES algorithm) that is constant for all 128 (Seed, DT) pairs. The DT value is incremented by 1 for each set. The seed values shall have no repeats within the set. The evaluator shall ensure that the values returned by the TSF match the expected values.

The evaluator shall perform a Monte Carlo Test. For this test, they supply an initial Seed and DT value to the TSF RBG function; each of these is 128 bits. The evaluator shall also provide a key (of the length appropriate to the AES algorithm) that is constant throughout the test. The evaluator then invoke the TSF RBG 10,000 times, with the DT value being incremented by 1 on each iteration, and the new seed for the subsequent iteration produced as specified in NIST-Recommended Random Number Generator Based on ANSI X9.31 Appendix A.2.4 Using the 3-Key Triple DES and AES Algorithms, Section 3. The evaluator shall ensure that the 10,000th value produced matches the expected value.

Implementations Conforming to NIST Special Publication 800-90

The evaluator shall perform 15 trials for the RBG implementation. If the RBG is configurable, the evaluator shall perform 15 trials for each configuration. The evaluator shall also confirm that the operational guidance contains appropriate instructions for configuring the RBG functionality.

If the RBG has prediction resistance enabled, each trial consists of (1) instantiate the deterministic RBG (DRBG), (2) generate the first block of random bits (3) generate a second block of random bits (4) uninstantiate. The evaluator verifies that the second block of random bits is the expected value. The evaluator shall generate eight input values for each trial. The first is a count (0 – 14). The next three are entropy input, nonce, and personalization string for the instantiate operation. The next two are additional input and entropy input for the first call to generate. The final two are additional input and entropy input for the second call to generate. These values are randomly generated. "Generate one block

of random bits" means to generate random bits with the number of returned bits equal to the Output Block Length (as defined in NIST SP 800-90).

If the RBG does not have prediction resistance, each trial consists of (1) instantiate DRBG, (2) generate the first block of random bits (3) reseed, (4) generate a second block of random bits (5) uninstantiate. The evaluator verifies that the second block of random bits is the expected value. The evaluator shall generate eight input values for each trial. The first is a count (0 – 14). The next three are entropy input, nonce, and personalization string for the instantiate operation. The fifth value is additional input to the first call to generate. The sixth and seventh are additional input and entropy input to the call to re-seed. The final value is additional input to the second generate call.

The following paragraphs contain more information on some of the input values to be generated/selected by the evaluator.

> ***Entropy input:*** *the length of the entropy input value must equal the seed length.*
>
> ***Nonce:*** *If a nonce is supported (CTR_DRBG with no derivation function (df) does not use a nonce), the nonce bit length is one-half the seed length.*
>
> ***Personalization string:*** *The length of the personalization string must be <= seed length. If the implementation only supports one personalization string length, then the same length can be used for both values. If more than one string length is supported, the evaluator shall use personalization strings of two different lengths. If the implementation does not use a personalization string, no value needs to be supplied.*
>
> ***Additional input:*** *the additional input bit lengths have the same defaults and restrictions as the personalization string lengths.*

FCS_TLS_EXT.1 Transport Level Security

FCS_TLS_EXT.1.1 The [selection, choose at least one of: VoIP client application, client device platform] shall implement one or more of the following protocols [selection: TLS 1.0 (RFC 2246), TLS 1.1 (RFC 4346), TLS 1.2 (RFC 5246)] using mutual authentication with certificates and supporting the following ciphersuites:

Mandatory Ciphersuites in accordance with RFC 3268:

- TLS_RSA_WITH_AES_128_CBC_SHA

Optional Ciphersuites: [*selection:*

- *TLS_RSA_WITH_AES_256_CBC_SHA*
- *TLS_DHE_RSA_WITH_AES_128_CBC_SHA*
- *TLS_DHE_RSA_WITH_AES_256_CBC_SHA*
- *TLS_RSA_WITH_AES_128_CBC_SHA256 as defined in RFC 5246*
- *TLS_RSA_WITH_AES_256_CBC_ SHA256 as defined in RFC 5246*
- *TLS_DHE_RSA_WITH_AES_128_CBC_ SHA256 as defined in RFC 5246*
- *TLS_DHE_RSA_WITH_AES_256_CBC_ SHA256 as defined in RFC 5246*
- *TLS_ECDHE_ECDSA_WITH_AES_128_GCM_SHA256 as defined in RFC 6460*
- *TLS_ECDHE_ECDSA_WITH_AES_256_GCM_SHA384 as defined in RFC 6460*
- *TLS_ECDHE_ECDSA_WITH_AES_128_CBC_SHA256 as defined in RFC 5289*

- *TLS_ECDHE_ECDSA_WITH_AES_256_CBC_SHA384 as defined in RFC 5289*
- [*assignment: Any other supported cipher suites*], *no other ciphersuite*]

Application Note:

The ciphersuites to be tested in the evaluated configuration are limited by this requirement. The ST author should select the optional ciphersuites that are supported; if there are no ciphersuites supported other than the mandatory suites, then "None" should be selected. It is necessary to limit the ciphersuites that can be used in an evaluated configuration administratively on the server in the test environment.

The Suite B algorithms (RFC 6460) listed above are the preferred algorithms for implementation. In addition, future publications of this PP will require Suite B algorithms. TLS 1.2 is the preferred protocol and may be required for EAP-TLS in the future.

FCS_TLS_EXT.1.2 The [selection, choose at least one of: VoIP client application, client device platform] shall not establish a trusted channel if the distinguished name (DN) contained in a certificate does not match the expected DN for the peer.

Application Note:
The DN may be in the Subject Name field or the Subject Alternative Name extension of the certificate. The expected DN may either be configured or may be compared to the Domain Name or IP address used by the peer.

Assurance Activity:
The evaluator shall check the description of the implementation of this protocol in the TSS to ensure that the ciphersuites supported are specified. The evaluator shall check the TSS to ensure that the ciphersuites specified are identical to those listed for this component. The evaluator shall also check the operational guidance to ensure that it contains instructions on configuring the TOE so that TLS conforms to the description in the TSS (for instance, the set of ciphersuites advertised by the TOE may have to be restricted to meet the requirements).
The evaluator shall verify that the TSS describes how the DN in the certificate is compared to the expected DN. If the DN is not compared automatically to the Domain Name or IP address, the evaluator shall ensure that the AGD guidance includes configuration of the expected DN for the connection.
Additional tests may be added in the future to test compliance with RFC 5246. The evaluator shall also perform the following tests:

- *Test 1: The evaluator shall establish a TLS connection using each of the ciphersuites specified by the requirement. This connection may be established as part of the establishment of a higher-level protocol, e.g., as part of an EAP session. It is sufficient to observe (on the wire by using a packet capture tool) the successful negotiation of a ciphersuite to satisfy the intent of the test; it is not necessary to examine the characteristics of the encrypted traffic in an attempt to discern the ciphersuite being used (for example, that the cryptographic algorithm is 128-bit AES and not 256-bit AES).*
- *Test 2: The following test is repeated for each supported certificate signing algorithm supported. The evaluator shall attempt to establish the connection using a server with a server certificate that contains the Server Authentication purpose in the extendedKeyUsage field and verify that a connection is established. The evaluator will then verify that the client rejects an otherwise valid server certificate that lacks the Server Authentication purpose in the extendedKeyUsage field and*

a connection is not established. Ideally, the two certificates should be identical except for the extendedKeyUsage field.

- *Test 3: The evaluator shall attempt a connection with a certificate where the DN matches either the configured expected DN or the Domain Name/IP address of the peer. The evaluator shall verify that the TSF is able to successfully connect. The evaluator shall attempt a connection with a certificate where the DN does not match either the configured expected DN or the Domain Name/IP address of the peer. The evaluator shall verify that the TSF is not able to successfully connect.*
- *Test 4: The evaluator shall configure the server to send a certificate in the TLS connection that the does not match the server-selected ciphersuite (for example, send a ECDSA certificate while using the TLS_RSA_WITH_AES_128_CBC_SHA ciphersuite or send a RSA certificate while using one of the ECDSA ciphersuites.) The evaluator shall verify that the TOE disconnects after receiving the server's Certificate handshake message.*
- *Test 5: The evaluator shall setup a man-in-the-middle tool between the TOE and the server and shall perform the following modifications to the traffic:*
    - *Modify at least one byte in the server's nonce in the Server Hello handshake message, and verify that the server denies the client's Finished handshake message.*
    - *Modify the server's selected ciphersuite in the Server Hello handshake message to be a ciphersuite not presented in the Client Hello handshake message. The evaluator shall verify that the client rejects the connection after receiving the Server Hello.*
    - *(conditional) If a DHE or ECDHE ciphersuite is supported, modify the signature block in the Server's KeyExchange handshake message, and verify that the client rejects the connection after receiving the Server KeyExchange.*
    - *Modify a byte in a CA field in the Server's Certificate Request handshake message. The modified CA field must not be the CA used to sign the client's certificate. The evaluator shall verify that the server rejects the connection after receiving the Client Finished handshake message.*
    - *Modify a byte in the Server Finished handshake message, and verify that the client sends a fatal alert upon receipt and does not send any application data.*

4.3.2 Identification and Authentication (FIA)

The baseline requirements for the TOE are fairly limited with respect to I&A, since no formal administrative or general purpose users are defined. The extent of the I&A required to be performed by the TOE relates to the authentication done at the machine level when establishing the TLS, and SDES/SRTP connections. Therefore, the requirements in this section cover only the credentials that are used by the protocols specified in this PP.

The certificates used by the TSF are those for the distant end TLS connection and the user's certificate (and associated private key).

FIA_X509_EXT.1 Extended: X509 Certificate Validation

FIA_X509_EXT.1.1 The [selection, choose at least one of: VoIP client application, client device platform] shall validate certificates in accordance with the following rules:

- RFC 5280 certificate validation and certificate path validation.

- Validate the certificate path by ensuring the basicConstraints extension is present and the cA flag is set to TRUE for all CA certificates.

- Validate the revocation status of the certificate using [selection: the Online Certificate Status Protocol (OCSP) as specified in RFC 2560, a Certificate Revocation List (CRL) as specified in RFC 5759].

- 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).

    - 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.

Application Note:
FIA_X509_EXT.1.1 lists the rules for validating certificates. The ST author shall select whether revocation status is verified using OCSP or CRLs. Certificates are used for peer authentication (FCS_TLS_EXT.1), trusted updates of TSF software (FPT_TUD_EXT.1) and optionally for integrity verification (FPT_TST_EXT.1), and if implemented, must be validated to contain the Code Signing purpose extendedKeyUsage. For TLS, certificates must be used to perform authentication and must be validated to contain the Server Authentication purpose extendedKeyUsage.

It should be noted that the validation is expected to end in a trusted root certificate.

FIA_X509_EXT.1.1 requires that the TOE perform certain checks on the certificate presented by a TLS server; namely that the extendedKeyUsage field of the server certificate includes "Server Authentication" and that the key agreement bit (for the Diffie-Hellman ciphersuites) or the key encipherment bit (for RSA ciphersuites) be set. Certificates obtained for use by the TOE will have to conform to these requirements.

FIA_X509_EXT.1.2 The [selection, choose at least one of: VoIP client application, client device platform] shall only treat a certificate as a CA certificate if the basicConstraints extension is present and the CA flag is set to TRUE.

Application Note:
This requirement applies to certificates that are used and processed by the VoIP Client Application or platform.

Assurance Activity:
The evaluator shall ensure the TSS describes where the check of validity of the certificates takes place – the TOE or the TOE platform. It may be that the TOE requests the platform to perform the check and provide a result, or the TOE may do the check itself. The evaluator ensures the TSS also provides a description of the certificate path validation algorithm, ensuring that it describes how the validation chain will terminate in a trusted root certificate.

The evaluator ensures the guidance documentation provides the user with the necessary information to setup the validation check whether it is done by the TOE or TOE platform. The guidance documentation

provides instructions how to select the method used for checking, as well as how to setup a protected communication path with the entity providing the information pertaining to certificate validity.

Regardless of the selection of "TOE" or "TOE Platform, the evaluator shall perform the following tests. This testing may be combined with the testing performed in the assurance activities for FCS_TLS_EXT.1 and FIA_X509_EXT.2.

- *Test 1: The evaluator shall demonstrate that validating a certificate without a valid certification path results in the function (application validation, trusted channel setup, or trusted software update) failing. The evaluator shall then load a certificate or certificates needed to validate the certificate to be used in the function, and demonstrate that the function succeeds. The evaluator then shall delete one of the certificates, and show that the function fails.*

- *Test 2: The evaluator shall demonstrate that validating an expired certificate results in the function failing.*

- *Test 3: The evaluator shall test that the TOE can properly handle revoked certificates – conditional on whether CRL or OCSP is selected; if both are selected, and then a test is performed for each method. The evaluator has to only test one up in the trust chain (future revisions may require to ensure the validation is done up the entire chain). The evaluator shall ensure that a valid certificate is used, and that the validation function succeeds. The evaluator then attempts the test with a certificate that will be revoked (for each method chosen in the selection) to ensure when the certificate is no longer valid that the validation function fails.*

- *Test 4: The evaluator shall construct a certificate path, such that the certificate of the CA issuing the TOE's certificate does not contain the basicConstraints extension. The validation of the certificate path fails.*

- *Test 5: The evaluator shall construct a certificate path, such that the certificate of the CA issuing the TOE's certificate has the cA flag in the basicConstraints extension not set. The validation of the certificate path fails.*

- *Test 6: The evaluator shall construct a certificate path, such that the certificate of the CA issuing the TOE's certificate has the cA flag in the basicConstraints extension set to TRUE. The validation of the certificate path succeeds.*

FIA_X509_EXT.2 Extended: X509 Certificate Use and Management

FIA_X509_EXT.2.1 The [selection, choose at least one of: VoIP client application, client device platform] shall use X.509v3 certificates as defined by RFC 5280 to support authentication for *SDES/SRTP, TLS*, and [selection: code signing for software updates, code signing for software integrity verification, no additional uses].

Application Note:
Certificates may optionally be used for trusted updates of system software (FPT_TUD_EXT.1) and software integrity verification (FPT_TST_EXT.1). If any of the code signing uses are selected then FIA_X509_EXT.2(2) must be included in the main body.

Each VoIP client application will have a unique X.509v3 certificate for use; the certificate is not to be reused among clients.

Assurance Activity:
Assurance activities for this element are tested through assurance activities for FCS_TLS_EXT.1, (conditionally) FPT_TUD_EXT.1 and FPT_TST_EXT.1.

FIA_X509_EXT.2.2 When the [selection, choose at least one of: VoIP client application, client device platform] cannot establish a connection to determine the validity of a certificate, the [selection, choose at least one of: VoIP client Application, client device platform] shall [selection: *allow the administrator to choose whether to establish or not establish the trusted channel in these cases, accept the certificate, not accept the certificate*].

Application Note:
Often a connection must be established to perform a verification of the revocation status of a certificate - either to download a CRL or to perform OCSP. The selection is used to describe the behavior in the event that such a connection cannot be established (for example, due to a network error). If the TOE has determined the certificate is valid according to all other rules in FIA_X509_EXT.1, the behavior indicated in the second selection shall determine the validity. The TOE must not accept the certificate if it fails any of the other validation rules in FIA_X509_EXT.1. If the administrator-configured option is selected by the ST Author, the ST Author must also select the appropriate function in FMT_SMF.1.

Assurance Activity:
The evaluator shall check the TSS to ensure that it describes how the TOE/platform chooses which certificates to use, and any necessary instructions in the administrative guidance for configuring the operating environment so that the TOE/platform can use the certificates. If this functionality is implemented entirely by the platform, the operational guidance for the TOE shall reference the applicable guidance for each platform.

The evaluator shall examine the TSS to confirm that it describes the behavior of the TOE/platform when a connection cannot be established during the validity check of a certificate used in establishing a trusted channel. If the requirement that the administrator is able to specify the default action, then the evaluator shall ensure that the operational guidance contains instructions on how this configuration action is performed. If this behavior is implemented entirely by the platform, the evaluator shall examine the ST of each platform to confirm that the selections for this element are contained in each platform's ST.

If this requirement is fully or partially implemented by the TOE, the evaluator shall perform Test 1 for each function in the system that requires the use of certificates:

- *Test 1: The evaluator shall demonstrate that using a valid certificate that requires certificate validation checking to be performed in at least some part by communicating with a non-TOE IT entity. The evaluator shall then manipulate the environment so that the TOE is unable to verify the validity of the certificate, and observe that the action selected in FIA_X509_EXT.2.2 is performed. If the selected action is administrator-configurable, then the evaluator shall follow the operational guidance to determine that all supported administrator-configurable options behave in their documented manner.*

FIA_X509_EXT.2.3 The [selection, choose at least one of: VoIP client Application, client device platform] shall not establish a trusted communication channel if the peer certificate is deemed invalid.

Application Note:
Trusted communication channels include any of SDES/SRTP, TLS performed by the TSF. Validity is determined by the certificate path, the expiration date, the revocation status in accordance with RFC 5280, and the distinguished name (DN) contained in the certificate.

Assurance Activity:
The evaluator shall check the TSS to ensure that it describes how the TOE chooses which certificates to use, and any necessary instructions in the administrative guidance for configuring the operating environment so that the TOE can use the certificates.

The evaluator shall examine the TSS to confirm that it describes the behavior of the TOE when a connection cannot be established during the validity check of a certificate used in establishing a trusted channel. If the requirement that the administrator is able to specify the default action, then the evaluator shall ensure that the operational guidance contains instructions on how this configuration action is performed.

The evaluator shall perform Test 1 for each function in the system that requires the use of certificates:

- *Test 1: The evaluator shall demonstrate that using a certificate without a valid certification path results in the function failing. Using the administrative guidance, the evaluator shall then load a certificate or certificates needed to validate the certificate to be used in the function, and demonstrate that the function succeeds. The evaluator then shall delete one of the certificates, and show that the function fails.*

The evaluator shall perform Test 2 for each function in the system that establishes a trusted path and requires the use of certificates:

- *Test 2: The evaluator shall demonstrate that using a valid certificate that requires certificate validation checking to be performed in at least some part by communicating with a non-TOE IT entity. The evaluator shall then manipulate the environment so that the TOE is unable to verify the validity of the certificate, and observe that the action selected in FIA_X509_EXT.2.2 is performed. If the selected action is administrator-configurable, then the evaluator shall follow the operational guidance to determine that all supported administrator-configurable options behave in their documented manner.*

- *Test 3: The evaluator shall use a certificate signed using the RSA or ECDSA algorithm to authenticate the TOE to the SIP Server during establishment of the trusted channel. This test ensures the TOE has the certificate for the trusted CA that signed the SIP Server's certificate and it will do a bit-wise comparison on the DN. This bit-wise comparison of the DN ensures that not only does the SIP Server have a certificate signed by the trusted CA, but the certificate is from the DN that is expected. The evaluator will configure the TSS to associate a certificate or DN (e.g., a certificate map in some implementations) with a trusted channel connection. This is what the DN is checked against.*

- *Test 4: The evaluator shall test that given a signed certificate from a trusted CA, that when the DN does not match – any of the four fields can be modified such that they do not match the expected value, that a trusted channel does not get established.*

- *Test 5: The evaluator shall ensure that the TOE is configurable to either establish a trusted channel, or not establish a trusted channel if a connection to the certificate validation entity cannot be reached. For each method selected for certificate validation, the evaluator attempts to validate the certificate – for the purposes of this test, it does not matter if the certificate is revoked or not. For the "mode" where a trusted channel is allowed to be established, the connection is made. Where the channel is not to be established, the connection is refused.*

Additional testing to ensure the requirements are satisfied is performed in conjunction with the requirements in FTP_ITC.

4.3.3 Security Management (FMT)

The TOE is not required to maintain a separate management role. It is, however, required to provide functionality to configure certain aspects of TOE operation that should not be available to the general user population.

FMT_SMF.1 Specification of Management Functions

FMT_SMF.1.1 The [selection, choose at least one of: VoIP client Application, client device platform] shall be capable of performing the following management functions:

- Configure cryptographic algorithms associated with protocols mandated in this PP,
- Load X5.09v3 certificates used for security functions in this PP,
- Configure certificate revocation check,
- Ability to update the TOE, and to verify the updates
- Ability to configure all security management functions identified in other sections of this PP,
- [selection: action taken when connection to verify validity of certificate cannot be established, [assignment: any additional management functions], no other actions].

Application Note:

For installation, the VoIP client application relies on the IT environment to authenticate the administrator to the client machine.

There may be some instances where a SIP Server "pushes" configuration information down to the VoIP client application. This is an acceptable form of management; the ST Author simply must make clear in the ST what management functions are performed by the VoIP client application, and which are performed by the SIP Server. It may be the case that the functions overlap (i.e., can be done by an end-user on the VoIP client application or by the SIP Server) and this is fine as long as the ST is clear and the guidance documentation describes how to perform the functions.

Assurance Activity:

The evaluator shall check to make sure that every management function mandated by the PP is described in the operational guidance and that the description contains the information required to perform the management duties associated with the management function. The evaluator shall test the

TOE's ability to provide the management functions by configuring the TOE and testing each option listed in the requirement above.

As stated in the application note, a TOE may be configured either locally or remotely by a SIP Server. The ST will clearly state which functions can be performed locally and remotely. The guidance documentation will describe how this is performed as well. The evaluator is expected to test this functions in all the ways in which the ST and guidance documentation state the configuration can be managed.

Note that the testing here may be accomplished in conjunction with the testing of other requirements, such as FCS_TLS_EXT.1.

4.3.4 Protection of the TSF (FPT)

FPT_TST_EXT.1 Extended: TSF Self Test

FPT_TST_EXT.1.1 The [selection, choose at least one of: VoIP Client Application, client device platform] shall run a suite of self tests during initial start-up (on power on) to demonstrate correct operation of the TSF.

FPT_TST_EXT.1.2 The [selection, choose at least one of: VoIP Client Application, client device 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 TSF-provided cryptographic services.

Application Note:

While the TOE is typically a software package running in the IT Environment, it is still capable of performing the self-test activities required above. It should be understood, however, that there is a significant dependency on the host environment in assessing the assurance provided by the tests mentioned above (meaning that if the host environment is compromised, the self tests will not be meaningful).

Assurance Activity:

The evaluator shall examine the TSS to ensure that it details the self tests that are run by the TSF on start-up; this description should include an outline of what the tests are actually doing (e.g., rather than saying "memory is tested", a description similar to "memory is tested by writing a value to each memory location and reading it back to ensure it is identical to what was written" shall be used). The evaluator shall ensure that the TSS makes an argument that the tests are sufficient to demonstrate that the TSF is operating correctly.

The evaluator shall examine the TSS to ensure that it describes how to verify the integrity of stored TSF executable code when it is loaded for execution. The evaluator shall ensure that the TSS makes an argument that the tests are sufficient to demonstrate that the integrity of stored TSF executable code has not been compromised. The evaluator also ensures that the TSS (or the operational guidance) describes the actions that take place for successful (e.g. hash verified) and unsuccessful (e.g., hash not verified) cases. The evaluator shall perform the following tests:

- *Test 1: The evaluator performs the integrity check on a known good TSF executable and verifies that the check is successful.*
- *Test 2: The evaluator modifies the TSF executable, performs the integrity check on the modified TSF executable and verifies that the check fails.*

FPT_TUD_EXT.1 Extended: Trusted Update

FPT_TUD_EXT.1.2 The [selection, choose at least one of: VoIP client application, client device platform] shall provide authorized administrators the ability to initiate updates to the TOE firmware/software.

FPT_TUD_EXT.1.3 The [selection, choose at least one of: VoIP client application, client device platform] shall provide a means to verify firmware/software updates to the TOE using a digital signature mechanism and [selection: published hash, no other functions] prior to installing those updates.

Application Note:

The digital signature mechanism referenced in the third element is the one specified in FCS_COP.1(2). The published hash referenced is generated by one of the functions specified in FCS_COP.1(3).

Assurance Activity:

Updates to the TOE are signed by an authorized source and may also have a hash associated with them, or are signed by an authorized source. If digital signatures are used, the definition of an authorized source is contained in the TSS, along with a description of how the certificates used by the update verification mechanism are contained on the device. The evaluator ensures this information is contained in the TSS. The evaluator also ensures that the TSS (or the operational guidance) describes how the candidate updates are obtained; the processing associated with verifying the digital signature or calculating the hash of the updates; and the actions that take place for successful (hash or signature was verified) and unsuccessful (hash or signature could not be verified) cases.

- *Test 1: The evaluator performs the version verification activity to determine the current version of the product. The evaluator obtains a legitimate update using procedures described in the operational guidance and verifies that it is successfully installed on the TOE. Then, the evaluator performs a subset of other assurance activity tests to demonstrate that the update functions as expected. After the update, the evaluator performs the version verification activity again to verify the version correctly corresponds to that of the update.*
- *Test 2: The evaluator performs the version verification activity to determine the current version of the product. The evaluator obtains or produces an illegitimate update, and attempts to install it on the TOE. The evaluator verifies that the TOE rejects the update.*

4.3.5 Trusted Path/Channel (FTP)

FTP_ITC.1(2) Inter-TSF Trusted Channel (TLS/SIP)

FTP_ITC.1.1(2) **Refinement:** The selection, choose at least one of: VoIP Client Application, client device platform] shall provide a communication channel between itself and **a SIP Server using TLS and no other protocol as specified in FCS_TLS_EXT.1 only** that is logically distinct from other communication channels and provides assured identification of its end points and protection of the channel data from modification and disclosure.

FTP_ITC.1.2(2) The selection, choose at least one of: VoIP Client Application, client device platform] shall permit the TSF to initiate communication via the trusted channel.

FTP_ITC.1.3(2) The selection, choose at least one of: VoIP Client Application, client device platform] shall initiate communication via the trusted channel for [all communications with the SIP server].

Application Note:

The TOE will establish a connection with the SIP server on start-up, and this will persist as long as the device is powered on and able to send/receive calls. The TOE is required to be able to use TLS to establish this connection.

***Assurance Activity**:*

The evaluator shall check the TSS section to confirm that it describes how this requirement is implemented in the TOE.

- *Test 1: The following test is repeated for each supported certificate signing algorithm supported. The evaluator shall attempt to establish a connection using a SIP server with an authentication server certificate that contains the Server Authentication purpose in the extendedKeyUsage field and verify that a connection is established. The evaluator will then verify that the client rejects an otherwise valid server certificate that lacks the Server Authentication purpose in the extendedKeyUsage field and a connection is not established. Ideally, the two certificates should be identical except for the extendedKeyUsage field.*
- *Test 2: The following test is repeated for each supported certificate signing algorithm supported. The evaluator shall verify that the TSF will only use a certificate that contains the Client Authentication purpose in the extendedKeyUsage field and verify that a connection is established. The evaluator will then verify that the TSF rejects an otherwise valid client certificate that lacks the Client Authentication purpose in the extendedKeyUsage field and a connection is not established. Ideally, the two certificates should be identical except for the extendedKeyUsage field.*

4.4 Security Assurance Requirements

The Security Objectives for the TOE in Section 3 were constructed to address threats identified in Section 2. The Security Functional Requirements (SFRs) in Section 4.2 are a formal instantiation of the Security Objectives.

While this section contains the complete set of SARs from the CC, the Assurance Activities to be performed by an evaluator are detailed both in Section 4.2 and Section 4.3 as well as in this section.

For each family, "Developer Notes" are provided on the developer action elements to clarify what, if any, additional documentation/activity needs to be provided by the developer. For the content/presentation and evaluator activity elements, additional assurance activities (to those already contained in Sections 4.2 and 4.3) are described as a whole for the family, rather than for each element. Additionally, the assurance activities described in this section are complementary to those specified in Sections 4.2 and 4.3.

The TOE security assurance requirements, summarized in Table 1, identify the management and evaluative activities required to address the threats identified in Section 2 of this PP.

Table 1: TOE Security Assurance Requirements

Assurance Class	Assurance Components	Assurance Components Description
Development	ADV_FSP.1	Basic Functional Specification
Guidance Documents	AGD_OPE.1	Operational User Guidance
	AGD_PRE.1	Preparative User Guidance
Tests	ATE_IND.1	Independent Testing - Conformance
Vulnerability Assessment	AVA_VAN.1	Vulnerability Analysis
Life Cycle Support	ALC_CMC.1	Labeling of the TOE
	ALC_CMS.1	TOE CM Coverage

4.4.1 Class ADV: Development

For TOEs conforming to this PP, the information about the TOE is contained in the TOE Summary Specification (TSS) portion of the ST guidance as well as documentation available to the end user. While it is not required that the TOE developer write the TSS, the TOE developer must concur with the description of the product that is contained in the TSS as it relates to the functional requirements. The Assurance Activities contained in Sections 4.2 and 4.3 should provide the ST authors with sufficient information to determine the appropriate content for the TSS section.

4.4.1.1 ADV_FSP.1 Basic functional specification

The functional specification describes the TOE Security Functionality Interfaces (TSFIs). It is not necessary to have a formal or complete specification of these interfaces. Additionally, because TOEs conforming to this PP will necessarily have interfaces to the Operational Environment that are not directly invokable by TOE users, there is little point specifying that such interfaces be described since only indirect testing of such interfaces may be possible. For this PP, the activities for this family should focus on understanding the interfaces presented in the TSS in response to the functional requirements and the interfaces presented in the AGD documentation. No additional "functional specification" documentation is necessary to satisfy the assurance activities specified.

The interfaces that need to be evaluated are characterized through the information needed to perform the assurance activities listed, rather than as an independent, abstract list.

Developer action elements:

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.

Developer Note: As indicated in the introduction to this section, the functional

specification is comprised of the information contained in the AGD_OPR and AGD_PRE documentation, coupled with the information provided in the TSS of the ST. The assurance activities in the functional requirements point to evidence that should exist in the documentation and TSS section. Since these are directly associated with the SFRs, the tracing in element ADV_FSP.1.2D is implicitly already done and no additional documentation is necessary.

Content and presentation elements:

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.

Evaluator action elements:

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.

Assurance Activity:

There are no specific assurance activities associated with these SARs. The functional specification documentation is provided to support the evaluation activities described in Sections 4.2 and 4.3, and other activities described for AGD, ATE, and AVA SARs. The requirements on the content of the functional specification information is implicitly assessed by virtue of the other assurance activities being performed; if the evaluator is unable to perform an activity because the there is insufficient interface information, then an adequate functional specification has not been provided.

4.4.2 Class AGD: Guidance Documents

The guidance documents will be provided with the developer's security target. Guidance must include a description of how the authorized user verifies that the Operational Environment can fulfill its role for the security functionality. The documentation should be in an informal style and readable by an authorized user.

Guidance must be provided for every operational environment that the product supports as claimed in the ST. This guidance includes:

- instructions to successfully install the TOE in that environment; and
- instructions to manage the security of the TOE as a product and as a component of the larger operational environment; and
- instructions to provide a protected administrative capability through the use of either TOE capabilities, environmental capabilities, or a combination of the two.

Guidance pertaining to particular security functionality must also be provided; specific requirements on such guidance are contained in the assurance activities specified in Sections 4.2 and 4.3.

4.4.2.1 AGD_OPE.1 Operational User Guidance

Developer action elements:

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

Developer Note: Rather than repeat information here, the developer should review the assurance activities for this component to ascertain the specifics of the guidance that the evaluators will be checking for. This will provide the necessary information for the preparation of acceptable guidance.

Content and presentation elements:

AGD_OPE.1.1C The operational user guidance shall describe what the authorized 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 the authorized user, 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 the authorized user, 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 the authorized user, 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 the authorized user, 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.

Evaluator action elements:

AGD_OPE.1.1E The evaluator *shall confirm* that the information provided meets all requirements for content and presentation of evidence.

Application Note:
The operational user guidance does not have to be contained in a single document. Guidance to users, administrators and application developers can be spread among documents or web pages. Where appropriate, the guidance documentation is expressed in the eXtensible Configuration Checklist Description Format (XCCDF) to support security automation.

Rather than repeat information here, the developer should review the assurance activities for this component to ascertain the specifics of the guidance that the evaluator will be checking for. This will provide the necessary information for the preparation of acceptable guidance.

Assurance Activity:
With respect to the management functions, while several have also been described in Sections 4.2 and 4.3, additional information is required as follows.

For TOEs that implement a cryptographic engine, the operational guidance shall contain instructions for configuring the cryptographic engine associated with the evaluated configuration of the TOE. It shall provide a warning that use of other cryptographic engines was not evaluated nor tested during the CC evaluation of the TOE.

The documentation must describe the process for verifying updates to the TOE, either by checking the hash or by verifying a digital signature. The evaluator shall verify that this process includes the following steps:

- *For hashes, a description of where the hash for a given update can be obtained. For digital signatures, instructions for obtaining the certificate that will be used by the FCS_COP.1(2) mechanism to ensure that a signed update has been received from the certificate owner. This may be supplied with the product initially, or may be obtained by some other means.*

- *Instructions for obtaining the update itself. This should include instructions for making the update accessible to the TOE (e.g., placement in a specific directory).*

Instructions for initiating the update process, as well as discerning whether the process was successful or unsuccessful. This includes generation of the hash/digital signature.

The operational guidance shall contain instructions for specifying the ports used for SRTP.

4.4.2.2 AGD_PRE.1 Preparative procedures

Developer action elements:

AGD_PRE.1.1D The developer shall provide the TOE including its preparative procedures.

Developer Note: As with the operational guidance, the developer should look to the assurance activities to determine the required content with respect to preparative procedures.

Content and presentation elements:

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.

Evaluator action elements:

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.

Assurance Activity:

As indicated in the introduction above, there are significant expectations with respect to the documentation—especially when configuring the operational environment to support TOE functional requirements. The evaluator shall check to ensure that the guidance provided for the TOE adequately addresses all platforms and components (that is, combination of hardware and operating system) claimed for the TOE in the ST.

The evaluator shall check to ensure that the following guidance is provided:

- *As indicated in the introductory material, administration of the TOE is performed by one or more administrators that are a subset of the group of all users of the TOE. While it must be the case that the overall system (TOE plus Operational Environment) provide this capability, the responsibility for the implementation of the functionality can vary from totally the Operational Environment's responsibility to totally the TOE's responsibility. At a high level, the guidance must contain the appropriate instructions so that the Operational Environment is configured so that it provides the portion of the capability for which it is responsible. If the TOE provides no mechanism to allow separation of administrative users from the population of users, then the instructions, for instance, would cover the OS configuration of the OS I&A mechanisms to provide a unique (OS-based) identity for users, and further guidance would instruct the installer on the configuration of the DAC mechanisms of the OS using the TOE administrative identity (or identities) so that only TOE administrators would have access to the administrative executables. If the TOE provides some or all of this functionality, then the appropriate requirements are*

included in the ST from Annex C, and the assurance activities associated with those requirements provide details on the guidance necessary for both the TOE and Operational Environment.

The evaluators shall also perform the following tests:

- *Test 1 [Conditional]: If the separation of administrative users from all TOE users is performed exclusively through the configuration of the Operational Environment, the evaluators will, for each configuration claimed in the ST, ensure that after configuring the system according to the administrative guidance, non-administrative users are unable to access TOE administrative functions.*

4.4.3 Class ATE: Tests

Testing is specified for functional aspects of the system as well as aspects that take advantage of design or implementation weaknesses. The former is done through ATE_IND family, while the latter is through the AVA_VAN family. At the assurance level specified in this PP, testing is based on advertised functionality and interfaces with dependency on the availability of design information. One of the primary outputs of the evaluation process is the test report as specified in the following requirements.

4.4.3.1 ATE_IND.1 Independent testing - Conformance

Testing is performed to confirm the functionality described in the TSS as well as the administrative (including configuration and operation) documentation provided. The focus of the testing is to confirm that the requirements specified in Sections 4.2 and 4.3 are being met, although some additional testing is specified for SARs in Section 4.4. The Assurance Activities identify the additional testing activities associated with these components. The evaluator produces a test report documenting the plan for and results of testing, as well as coverage arguments focused on the platform/TOE combinations that are claiming conformance to this PP.

	Developer action elements:
ATE_IND.1.1D	The developer shall provide the TOE for testing.
	Content and presentation elements:
ATE_IND.1.1C	The TOE shall be suitable for testing.
	Evaluator action elements:
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.

Assurance Activity:

The evaluator shall prepare a test plan and report documenting the testing aspects of the system. The test plan covers all of the testing actions contained in the CEM and the body of this PP's Assurance

Activities. While it is not necessary to have one test case per test listed in an Assurance Activity, the evaluators must document in the test plan that each applicable testing requirement in the ST is covered.

The Test Plan identifies the platforms to be tested, and for those platforms not included in the test plan but included in the ST, the test plan provides a justification for not testing the platforms. This justification must address the differences between the tested platforms and the untested platforms, and make an argument that the differences do not affect the testing to be performed. It is not sufficient to merely assert that the differences have no affect; rationale must be provided. If all platforms claimed in the ST are tested, then no rationale is necessary.

The test plan describes the composition of each platform to be tested, and any setup that is necessary beyond what is contained in the AGD documentation. It should be noted that the evaluators are expected to follow the AGD documentation for installation and setup of each platform either as part of a test or as a standard pre-test condition. This may include special test drivers or tools. For each driver or tool, an argument (not just an assertion) is provided that the driver or tool will not adversely affect the performance of the functionality by the TOE and its platform. This also includes the configuration of the cryptographic engine to be used. The cryptographic algorithms implemented by this engine are those specified by this PP and used by the cryptographic protocols being evaluated (SDES, and TLS).

The test plan identifies high-level test objectives as well as the test procedures to be followed to achieve those objectives. These procedures include expected results. The test report (which could just be an annotated version of the test plan) details the activities that took place when the test procedures were executed, and includes the actual results of the tests. This shall be a cumulative account, so if there was a test run that resulted in a failure; a fix installed; and then a successful re-run of the test, the report would show a "fail" and "pass" result (and the supporting details), and not just the "pass" result.

4.4.4 Class AVA: Vulnerability assessment

For the first generation of this protection profile, the evaluation lab is expected to survey open sources to learn what vulnerabilities have been discovered in these types of products. In most cases, these vulnerabilities will require sophistication beyond that of a basic attacker. Until penetration tools are created and uniformly distributed to the evaluation labs, evaluators will not be expected to test for these vulnerabilities in the TOE. The labs will be expected to comment on the likelihood of these vulnerabilities given the documentation provided by the vendor. This information will be used in the development of penetration testing tools and for the development of future protection profiles.

4.4.4.1 AVA_VAN.1 Vulnerability survey

Developer action elements:

AVA_VAN.1.1D The developer shall provide the TOE for testing.

Content and presentation elements:

AVA_VAN.1.1C The TOE shall be suitable for testing.

Evaluator action elements:

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.

Assurance Activity:

*As with ATE_IND the evaluator shall generate a report to document their findings with respect to this requirement. This report could physically be part of the overall test report mentioned in ATE_IND, or a separate document. The evaluator performs a search of public information to determine the vulnerabilities that have been found in Mobility {*mobility component*} in general, as well as those that pertain to the particular TOE. The evaluator documents the sources consulted and the vulnerabilities found in the report. For each vulnerability found, the evaluator either provides a rationale with respect to its non-applicability or the evaluator formulates a test (using the guidelines provided in ATE_IND) to confirm the vulnerability, if suitable. Suitability is determined by assessing the attack vector needed to take advantage of the vulnerability. For example, if the vulnerability can be detected by pressing a key combination on boot-up, a test would be suitable at the assurance level of this PP. If exploiting the vulnerability requires expert skills and an electron microscope, for instance, then a test would not be suitable and an appropriate justification would be formulated.*

4.4.5 Class ALC: Life-cycle support

At the assurance level provided for TOEs conformant to this PP, life-cycle support is limited to end-user-visible aspects of the life-cycle, rather than an examination of the TOE vendor's development and configuration management process. This is not meant to diminish the critical role that a developer's practices play in contributing to the overall trustworthiness of a product; rather, it's a reflection on the information to be made available for evaluation at this assurance level.

4.4.5.1 ALC_CMC.1 Labeling of the TOE

This component is targeted at identifying the TOE such that it can be distinguished from other products or version from the same vendor and can be easily specified when being procured by an end user.

Developer action elements:

ALC_CMC.1.1D The developer shall provide the TOE and a reference for the TOE.

Content and presentation elements:

ALC_CMC.1.1C The TOE shall be labeled with its unique reference.

Evaluator action elements:

ALC_CMC.2.1E The evaluator *shall confirm* that the information provided meets all requirements for content and presentation of evidence.

Assurance Activity:

The evaluator shall check the ST to ensure that it contains an identifier (such as a product name/version number) that specifically identifies the version that meets the requirements of the ST. Further, the evaluator shall check the AGD guidance and TOE samples received for testing to ensure that the version number is consistent with that in the ST. If the vendor maintains a web site advertising the TOE, the evaluator shall examine the information on the web site to ensure that the information in the ST is sufficient to distinguish the product.

4.4.5.2 ALC_CMS.1 TOE CM coverage

Given the scope of the TOE and its associated evaluation evidence requirements, this component's assurance activities are covered by the assurance activities listed for ALC_CMC.1.

Developer action elements:

ALC_CMS.2.1D The developer shall provide a configuration list for the TOE.

Content and presentation elements:

ALC_CMS.2.1C The configuration list shall include the following: the TOE itself; and the evaluation evidence required by the SARs.

ALC_CMS.2.2C The configuration list shall uniquely identify the configuration items.

Evaluator action elements:

ALC_CMS.2.1E The evaluator *shall confirm* that the information provided meets all requirements for content and presentation of evidence.

Assurance Activity:

The "evaluation evidence required by the SARs" in this PP is limited to the information in the ST coupled with the guidance provided to administrators and users under the AGD requirements. By ensuring that the TOE is specifically identified and that this identification is consistent in the ST and in the AGD guidance (as done in the assurance activity for ALC_CMC.1), the evaluator implicitly confirms the information required by this component.

RATIONALE

The rationale for choosing these security assurance requirements is that they represent a more objective and repeatable level of assurance than has previously been accomplished using the generic CEM process. These assurance activities are, however, based on activities in the CEM, but tailored to this technology and specifically to the functional requirement contained in this PP. The assurance requirements and activities represent sufficient documentation and testing to mitigate the threats and achieve the objectives presented in the PP. If vulnerabilities are found in these types of products, then more stringent security assurance requirements may be mandated based on actual vendor practices.

ANNEX A: SUPPORTING TABLES

In this Protection Profile, the focus in the initial sections of the document is to use a narrative presentation in an attempt to increase the overall understandability of the threats to network devices, the methods used to mitigate those threats, and the extent of the mitigation achieved by compliant TOEs. This presentation style does not readily lend itself to a formalized evaluation activity, so this Annex contains the tabular artifacts that can be used for the evaluation activities associated with this document.

Assumptions

The specific conditions listed in the following subsections are assumed to exist in the TOE's Operational Environment. These assumptions include both practical realities in the development of the TOE security requirements and the essential environmental conditions on the use of the TOE. If the TOE is placed in an operational environment that does not meet these assumptions, the TOE may no longer be able to provide all of its security functionality.

Table 2: TOE Assumptions

Assumption Name	Assumption Name
A.AVAILABILITY	Network resources shall be available to allow VoIP clients to satisfy mission requirements and to transmit information.
A.OPER_ENV	The operational environment of the TOE appropriately addresses those requirements, threats, and policies not applicable to the TOE itself, but that are necessary to support the correct operation of the TOE.
A.TRUSTED_CONFIG	Personnel configuring the TOE and its operational environment will follow the applicable security configuration guidance.

Threats

The following table lists the threats addressed by the VoIP client application and the operational environment. The assumed level of expertise of the attacker for all the threats identified below is unsophisticated.

Table 3: Threats

Threat	Description of Threat
T.TSF_CONFIGURATION	Failure to allow configuration of the TSF may prevent its users from being able to adequately implement their particular security policy, leading to a compromise of user information.
T.TSF_FAILURE	Security mechanisms of the TOE may fail, leading to a compromise of the TSF.
T.UNAUTHORIZED_ACCESS	A user may gain unauthorized access to the TOE data. A malicious user, process, or external IT entity may masquerade as an authorized entity in order to gain unauthorized access to data or TOE resources. A malicious user, process, or external IT entity may misrepresent itself as the TOE to obtain

	identification and authentication data.
T.UNAUTHORIZED_UPDATE	A malicious party attempts to supply the end user with an update to the product that may compromise the security features of the TOE.
T.USER_DATA_REUSE	Voice data may be inadvertently sent to a destination not intended because it is sent outside the voice call.

Security Objectives for the TOE

Table 4: Security Objectives for the TOE

Objective	Objective Description
O.PROTECTED_COMMUNICATIONS	The TOE will provide protected communication channels with authorized IT entities (SIP Server and other VoIP applications).
O.TSF_SELF_TEST	The TOE will provide the capability to test some subset of its security functionality to ensure it is operating properly.
O.VERIFIABLE_UPDATES	The TOE will provide the capability to help ensure that any updates to the TOE can be verified by the administrator to be unaltered and (optionally) from a trusted source.

The following table contains objectives for the Operational Environment. As assumptions are added to the PP, these objectives should be augmented to reflect such additions.

Table 5: Security Objectives for the Operational Environment

Objective	Objective Description
OE.AUTHORIZED_USER	The user of the TOE is non-hostile and follows all user guidance.
OE.OPER_ENV	The operational environment will provide a SIP infrastructure to establish a VoIP connection; a PKI to provide certificates; and an execution domain to support correct operation of the TOE.
OE.TRUSTED_ADMIN	TOE Administrators are trusted to follow and apply all administrator guidance in a trusted manner.

ANNEX B: Optional Requirements

As indicated in the introduction to this PP, the baseline requirements (those that must be performed by the TOE or its underlying platform) are contained in the body of this PP. Additionally, there are three other types of requirements specified in Annexes B, C, and D.

The first type (in this Annex) are requirements that can be included in the ST, but do not have to be in order for a TOE to claim conformance to this PP. The second type (in Annex C) are requirements based on selections in the body of the PP: if certain selections are made, then additional requirements in that appendix will need to be included. The third type (in Annex D) are components that are not required in order to conform to this PP, but will be included in the baseline requirements in future versions of this PP, so adoption by VPN Client vendors is encouraged. Note that the ST author is responsible for ensuring that requirements that may be associated with those in Annex B, Annex C, and/or Annex D but are not listed (e.g., FMT-type requirements) are also included in the ST.

There are no optional requirements at this time. In future versions, this section may contain additional requirements that the ST author will be responsible for including.

ANNEX C: Selection-Based Requirements

As indicated in the introduction to this PP, the baseline requirements (those that must be performed by the TOE or its underlying platform) are contained in the body of this PP. There are additional requirements based on selections in the body of the PP: if certain selections are made, then additional requirements below will need to be included.

Identification and Authentication (FIA)

FIA_X509_EXT.2(1) Extended: X509 Authentication

FIA_X509_EXT.2.4(1) The [selection, choose at least one of: VoIP client application, client device platform] shall not [selection: install, execute] code if the code signing certificate is deemed invalid.

Application Note:
Certificates may optionally be used for trusted updates of system software (FPT_TUD_EXT.1.3) and software integrity verification (FPT_TST_EXT.1.2). If any of the code signing uses is selected in FIA_X509_EXT.2.1, FIA_X509_EXT.2.4 must be included in the main body.

Assurance Activity:
The assurance activity for this requirement is performed in conjunction with the assurance activity for FIA_X509_EXT.1 and FIA_X509_EXT.2.

ANNEX D: Objective Requirements

As indicated in the introduction to this PP, the baseline requirements (those that must be performed by the TOE and the TOE or its platform) are contained in the body of this PP. There are additional requirements that specify security functionality that is desirable and these requirements are contained in this Annex. It is expected that these requirements will transition from objective requirements to baseline requirements in future versions of this PP.

At any time these may be included in the ST such that the TOE is still conformant to this PP.

Security Audit (FAU)
If audit generation is provided by the TOE, the following audit requirements must be included in the ST with the ST Author making the appropriate selections and assignments.

Security Audit Data Generation (FAU_GEN)
FAU_GEN.1 Audit Data Generation
FAU_GEN.1.1 The VoIP client application shall be able to generate an audit record of the following auditable events:

a) Start-up and shutdown of audit functions;
b) All administrative actions;
c) [specifically defined auditable events listed in Table 6].

Application Note:
The ST author can include other auditable events directly in the table; they are not limited to the list presented.
In the case of item 'a', the audit functions referred to are those provided by the TOE. For example, in the case that the TOE was a stand-alone executable, auditing the start-up and shutdown of the TOE itself would be sufficient to meet the requirements of this clause.
Many auditable aspects of SFRs included in this document deal with administrative actions. Item 'c' above requires all administrative actions to be auditable, so no additional specification of the auditability of these actions is present in Table 6. While the TOE itself does not need to provide the ability to perform I&A for an administrator, this requirement implies that the TOE possesses the capability to audit the events described in the PP as "administrative actions" (primarily dealing with configuration of the functionality provided by the TOE). It is expected that the AGD guidance details the steps needed to ensure the audit data generated by the TOE is integrated with the audit capabilities of the underlying IT environment.

Assurance Activity:

The evaluator shall check the operational guidance and ensure that it lists all of the auditable events and provides a format for audit records. Each audit record format type must be covered, along with a brief description of each field. The evaluator shall check to make sure that every audit event type mandated by the PP is described and that the description of the fields contains the information required in FAU_GEN.1.2, and the additional information specified in Table 6.

The evaluator shall in particular ensure that the operational guidance is clear in relation to the contents for failed cryptographic events. In Table 6, information detailing the cryptographic mode of operation and a name or identifier for the object being encrypted is required. The evaluator shall ensure that name

or identifier is sufficient to allow an administrator reviewing the audit log to determine the context of the cryptographic operation (for example, performed during a key negotiation exchange, performed when encrypting data for transit) as well as the non-TOE endpoint of the connection for cryptographic failures relating to communications with other IT systems.

The evaluator shall also make a determination of the administrative actions that are relevant in the context of this PP. The TOE may contain functionality that is not evaluated in the context of this PP because the functionality is not specified in an SFR. This functionality may have administrative aspects that are described in the operational guidance. Since such administrative actions will not be performed in an evaluated configuration of the TOE, the evaluator shall examine the operational guidance and make a determination of which administrative commands, including subcommands, scripts, and configuration files, are related to the configuration (including enabling or disabling) of the mechanisms implemented in the TOE that are necessary to enforce the requirements specified in the PP, which thus form the set of "all administrative actions". The evaluator may perform this activity as part of the activities associated with ensuring the AGD_OPE guidance satisfies the requirements.

The evaluator shall test the TOE's ability to correctly generate audit records by having the TOE generate audit records in accordance with the assurance activities associated with the functional requirements in this PP. Additionally, the evaluator shall test that each administrative action applicable in the context of this PP is auditable. When verifying the test results, the evaluator shall ensure the audit records generated during testing match the format specified in the administrative guide, and that the fields in each audit record have the proper entries.

Note that the testing here can be accomplished in conjunction with the testing of the security mechanisms directly. For example, testing performed to ensure that the administrative guidance provided is correct verifies that AGD_OPE.1 is satisfied and should address the invocation of the administrative actions that are needed to verify the audit records are generated as expected.

FAU_GEN.1.2 The [selection, choose at least one of: VoIP Client Application, client device platform] shall record within each audit record at least the following information:

- Date and time of the event;
- Type of event;
- Subject identity;
- The outcome (success or failure) of the event; and
- Additional information in Table 6.

Application Note:
As with the previous component, the ST author should update Table 6 with any additional information generated. "Subject identity" in the context of this requirement could either be the administrator's user id or the affected network interface, for example.

Assurance Activity:
This activity should be accomplished in conjunction with the testing of FAU_GEN.1.1.

Security Audit Event Selection (FAU_SEL)
FAU_SEL.1 Selective Audit
FAU_SEL.1.1 The VoIP Client Application shall be able to select the set of events to be audited from the set of all auditable events based on the following attributes:

a) Event type;

b) Success of auditable security events;
c) Failure of auditable security events; and
d) [assignment: other attributes].

Application Note:
The intent of this requirement is to identify all criteria that can be selected to trigger an audit event. This can be configured through an interface on the client for a user/administrator to invoke, or it could be an interface that the SIP Server uses to instruct the client on which events are to be audited. For the ST author, the assignment is used to list any additional criteria or "none". The auditable event types are listed in Table 6.

Assurance Activity:

The evaluator shall review the administrative guidance to ensure that the guidance itemizes all event types, as well as describes all attributes that are to be selectable in accordance with the requirement, to include those attributes listed in the assignment. The administrative guidance shall also contain instructions on how to set the pre-selection, or how the SIP Server will configure the client, as well as explain the syntax (if present) for multi-value pre-selection. The administrative guidance shall also identify those audit records that are always recorded, regardless of the selection criteria currently being enforced.

The evaluator shall also perform the following tests:

- *Test 1: For each attribute listed in the requirement, the evaluator shall devise a test to show that selecting the attribute causes only audit events with that attribute (or those that are always recorded, as identified in the administrative guidance) to be recorded.*
- *Test 2 [conditional]: If the TSF supports specification of more complex audit pre-selection criteria (e.g., multiple attributes, logical expressions using attributes) then the evaluator shall devise tests showing that this capability is correctly implemented. The evaluator shall also, in the test plan, provide a short narrative justifying the set of tests as representative and sufficient to exercise the capability.*

Table 6: Auditable Events

Requirement	Auditable Events	Additional Audit Record Contents
FAU_GEN.1	None.	
FAU_SEL.1	All modifications to the audit configuration that occur while the audit collection functions are operating.	None.
FCS_CKM.1	Failure of the key generation activity.	None.
FCS_CKM.2	None.	
FCS_CKM_EXT.4	Failure of the key zeroization process.	Identity of object or entity being cleared.
FCS_COP.1(1)	Failure of encryption or decryption.	Cryptographic mode of operation, name/identifier of object being encrypted/decrypted.
FCS_COP.1(2)	Failure of cryptographic signature.	Cryptographic mode of operation, name/identifier of object being signed/verified.

Requirement	Auditable Events	Additional Audit Record Contents
FCS_COP.1(3)	Failure of hashing function.	Cryptographic mode of operation, name/identifier of object being hashed.
FCS_COP.1(4)	Failure in Cryptographic Hashing for Non-Data Integrity.	Cryptographic mode of operation, name/identifier of object being hashed.
FCS_RBG_EXT.1	Failure of the randomization process.	None.
FCS_SRTP_EXT.1	Failure to establish a SDES/SRTP session. Establishment/termination of a SDES/SRTP session.	Reason for failure. Non-TOE endpoint of connection (IP address).
FCS_TLS_EXT.1	Failure to establish a TLS session. Establishment/termination of a TLS session.	Reason for failure. Non-TOE endpoint of connection (IP address).
FDP_VOP_EXT.1	None.	
FIA_SIPC_EXT.1	Session establishment with peer.	Source and destination address.
FIA_X509_EXT.1	None.	
FMT_SMF.1	None.	
FPT_TST_EXT.1	Execution of this set of TSF self-tests. Detected integrity violations.	For integrity violations, the TSF code file that caused the integrity violation.
FPT_TUD_EXT.1	Initiation of the update. Any failure to verify the integrity of the update.	No additional information.
FTP_ITC.1	All attempts to establish a trusted channel. Detection of modification of channel data.	Identification of the non-TOE endpoint of the channel.

Trusted Path/Channel (FTP)

FTP_ALT_EXT.1 Extended: Trusted Channel Alert

FTP_ALT_EXT.1 The VoIP Client Application shall provide a visual alert if a call is placed or received outside the secure channel specified in FTP_ITC.1.

Application Note:
This alert serves to notify the user that their voice call is not protected. Any manner of visual alert is acceptable, such as an unsecured lock, red banner, etc. This is applicable only to calls made when the VoIP application is running; the VoIP application does not have to monitor all calls in/out of the device that do not use the VoIP application/VoIP technology. If this visual alert is a configurable notification, the ST Author must include the selection in FMT_SMF.1.

***Assurance Activity**:*

The evaluator shall check the TSS to verify that it describes the circumstances in which the call is not protected and the method used to provide visual notification to the user. If this is a configurable notification, the TOE guidance shall contain the information necessary to properly configure this alert.

The evaluator shall perform the following tests:

- *Test 1: The evaluator shall ensure that the secure channel is up and will make a call both to and from the TOE. The evaluator shall verify that no visual notification is received.*
- *Test 2: The evaluator shall configure the environment into each of the circumstances described in the TSS and make an unprotected call both to and from the TOE. The evaluator shall verify that the visual alert is present.*

ANNEX E: Entropy Documentation and Assessment

The documentation of the entropy source should be detailed enough that, after reading, the evaluator will thoroughly understand the entropy source and why it can be relied upon to provide entropy. This documentation should include multiple detailed sections: design description, entropy justification, operating conditions, and health testing. This documentation is not required to be part of the TSS.

Design Description

Documentation shall include the design of the entropy source as a whole, including the interaction of all entropy source components. It will describe the operation of the entropy source to include how it works, how entropy is produced, and how unprocessed (raw) data can be obtained from within the entropy source for testing purposes. The documentation should walk through the entropy source design indicating where the random comes from, where it is passed next, any post-processing of the raw outputs (hash, XOR, etc.), if/where it is stored, and finally, how it is output from the entropy source. Any conditions placed on the process (e.g., blocking) should also be described in the entropy source design. Diagrams and examples are encouraged.

This design must also include a description of the content of the security boundary of the entropy source and a description of how the security boundary ensures that an adversary outside the boundary cannot affect the entropy rate.

Entropy Justification

There should be a technical argument for where the unpredictability in the source comes from and why there is confidence in the entropy source exhibiting probabilistic behavior (an explanation of the probability distribution and justification for that distribution given the particular source is one way to describe this). This argument will include a description of the expected entropy rate and explain how you ensure that sufficient entropy is going into the TOE randomizer seeding process. This discussion will be part of a justification for why the entropy source can be relied upon to produce bits with entropy.

Operating Conditions

Documentation will also include the range of operating conditions under which the entropy source is expected to generate random data. It will clearly describe the measures that have been taken in the system design to ensure the entropy source continues to operate under those conditions. Similarly, documentation shall describe the conditions under which the entropy source is known to malfunction or become inconsistent. Methods used to detect failure or degradation of the source shall be included.

Health Testing

More specifically, all entropy source health tests and their rationale will be documented. This will include a description of the health tests, the rate and conditions under which each health test is performed (e.g., at startup, continuously, or on-demand), the expected results for each health test, and rationale indicating why each test is believed to be appropriate for detecting one or more failures in the entropy source.

ANNEX F: Glossary

Administrator – a user that has administrative privilege to configure the TOE

Authorized – an entity granted access privileges to an object, system or system entity

Critical Security Parameter (CSP) – security related information, e.g. secret and private cryptographic keys, and authentication data such as passwords and PINs, whose disclosure or modification can compromise the security of a cryptographic module

Entropy Source – this cryptographic function provides a seed for a random number generator by accumulating the outputs from one or more noise sources. The functionality includes a measure of the minimum work required to guess a given output and tests to ensure that the noise sources are operating properly

FIPS-approved cryptographic function – a security function (e.g., cryptographic algorithm, cryptographic key management technique, or authentication technique) that is either: 1) specified in a Federal Information Processing Standard (FIPS), or 2) adopted in a FIPS and specified either in an appendix to the FIPS or in a document referenced by the FIPS

IT Environment – hardware and software that are outside the TOE boundary that support the TOE functionality and security policy

Operational Environment – the environment in which the TOE is operated

Public Network – a network that is visible to all users and entities and does not protect against unauthorized access (e.g. internet)

Security Assurance Requirement (SAR) – description of how assurance is to be gained that the TOE meets the SFR

Security Functional Requirement (SFR) – translation of the security objectives for the TOE into a standardized language

Security Target (ST) – implementation-dependent statement of security needs for a specific identified TOE

Target of Evaluation (TOE) – set of software, firmware and/or hardware possibly accompanied by guidance. For this PP the TOE is the VoIP Client Application

Threat Agent - an entity that tries to harm an information system through destruction, disclosure, modification of data, and/or denial of service

TOE Security Functionality (TSF) – combined functionality of all hardware, software, and firmware of a TOE that must be relied upon for the correct enforcement of the SFRs

TOE Summary Specification (TSS) – a description of how the TOE satisfies all of the SFRs

VoIP Client Application – the TOE, allows users to establish an encrypted SDES/SRTP tunnel across an unprotected public network for transmission of voice data

VoIP Client Application User – a user operating the TOE