Assurance Activities Report for a Target of Evaluation

VMware Workspace ONE Boxer Email Client Version 23.11

Assurance Activities Report (AAR) Version 1.0

April 24, 2024

Security Target (Version 1.0)

Evaluated by:

Booz | Allen | Hamilton

Booz Allen Hamilton Common Criteria Test Laboratory
NIAP Lab # 200423
1100 West St.
Laurel, MD 20707

Prepared for:
National Information Assurance Partnership
Common Criteria Evaluation and Validation Scheme

The Developer of the TOE:

VMware
1155 Perimeter Center West
Suite 100
Atlanta, GA 30338

The Author of the Security Target:

Booz Allen Hamilton,
1100 West St.
Laurel, 20707 USA

The TOE Evaluation was sponsored by:

VMware
1155 Perimeter Center West
Suite 100
Atlanta, GA 30338

Evaluation Personnel:

Herbert Markle
Christopher Rakaczky
Evan Seiz

Applicable Common Criteria Version

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

Common Evaluation Methodology Version

Common Criteria for Information Technology Security Evaluation, Evaluation Methodology, April 2017 Version 3.1 Revision 5

Table of Contents

1 Purpose

The purpose of this document is to serve as a non-proprietary attestation that this evaluation has satisfied all of the TSS, AGD, and ATE Assurance Activities required by the Protection Profiles/Extended Packages to which the TOE claims exact conformance. This will give system integrators valuable information about product configuration and testing, help to align Common Criteria evaluations with DISA Security Requirements Guides and Security Test Implementation Guides (SRGs/STIGs), and thereby streamline the process for U.S. Government procurement of validated products.

2 TOE Summary Specification Assurance Activities

The evaluation team completed the testing of the Security Target (ST) 'VMware Workspace ONE Boxer Email Client Version 23.11 Security Target v1.0' and confirmed that the TOE Summary Specification (TSS) contains all Assurance Activities as specified by the *Protection Profile for Application Software Version 1.4 [APP_PP]* and *Application Software Extended Package for Email Clients v2.0 [EC_EP]*. The evaluators were able to individually examine each SFR's TSS statements and determine that they comprised sufficient information to address each SFR claimed by the TOE as well as meet the expectations of the APP_PP and EC_EP Assurance Activities.

Through the evaluation of ASE_TSS.1-1, described in the ETR, the evaluators were able to determine that each SFR was described in enough detail to demonstrate that the TSF addresses the SFR. However, in some cases the Assurance Activities that are specified in the claimed source material instruct the evaluator to examine the TSS for a description of specific behavior to ensure that each SFR is described to an appropriate level of detail. The following is a list of each SFR, the TSS Assurance Activities specified for the SFR, and how the TSS meets the Assurance Activities. Additionally, each SFR is accompanied by the source material App PP v1.4 and Email Client EP v2.0 that defines where the most up-to-date TSS Assurance Activity was defined.

FCS_CKM_EXT.1.1 – TD0717 *"The evaluator shall inspect the application and its developer documentation to determine if the application needs asymmetric key generation services. If not, the evaluator shall verify the generate no asymmetric cryptographic keys selection is present in the ST. Otherwise, the evaluation activities shall be performed as stated in the selection-based requirements."*

This activity is considered satisfied as the TSS states in section 8.1.1 that the TOE invokes the platform to support asymmetric key generation in support of TLS communications.

FCS_CKM.1.1/AK – TD0717 "*The evaluator shall ensure that the TSS identifies the key sizes supported by the TOE. If the ST specifies more than one scheme, the evaluator shall examine the TSS to verify that it identifies the usage for each scheme.*

If the application invokes platform-provided functionality for asymmetric key generation, then the evaluator shall examine the TSS to verify that it describes how the key generation functionality is invoked."

This activity is considered satisfied as the TSS states in section 8.1.1 identifies that the TOE invokes the platform to support asymmetric key generation in support of TLS communications. The platform provided functionality support both RSA schemes using cryptographic key sizes of 2048-bit or greater that meet FIPS PUB 186-4, "Digital Signature Standard (DSS)", Appendix B.3 and ECC schemes using "NIST curves" P-256, P-384 that meet FIPS PUB 186-4, "Digital Signature Standard (DSS)", Appendix B.4

.

FCS_CKM.2.1 – *"The evaluator shall ensure that the supported key establishment schemes correspond to the key generation schemes identified in FCS_CKM.1.1. If the ST specifies more than one scheme, the evaluator shall examine the TSS to verify that it identifies the usage for each scheme."*

This activity is considered satisfied as the TSS states in section 8.1.2 that the TOE invokes the platform-provided functionality to perform Cryptographic Key Establishment for RSA and Elliptic curve-based key

establishment (ECC) which correspond to FCS_CKM.1 declarations. Section 6.3.1.2 for FCS_CKM.1.1/AK of the ST consistently states that the application shall invoke platform-provided functionality to generate asymmetric cryptographic keys in accordance with RSA and ECC schemes. This is applicable to the TOE being run on any of the iOS, iPadOS, and Android based devices. . - Pass

[EC_EP] FCS_CKM_EXT.3.1 – *"The evaluator shall verify the TSS for a high level description of method used to protect keys stored in non-volatile memory.*

The evaluator shall verify the TSS to ensure it describes the storage location of all keys and the protection of all keys stored in non-volatile memory. The description of the key chain shall be reviewed to ensure FCS_COP_EXT.2 is followed for the storage of wrapped or encrypted keys in non-volatile memory and plaintext keys in non-volatile memory meet one of the criteria for storage."

Section 8.1.3 of the TSS provides a pointer to Tables 17 and 18 for a high level description of the methods used to protect and destroy keys stored in non-volatile memory. Each table, one for Android and on for iOS/ iPadOS, contains a column to identify all of the keys, the storage location for volatile and non-volatile memory, a high level description of when and how the keys are destroyed, and a description of use each key. These tables are also referenced by Section 8.1.10 (FCS_COP_EXT.2.1(Android), FCS_COP_EXT.2.1(iOS) & FCS_COP_EXT.2.1(iPadOS)) which identifies Android Boxer implements NIST SP 800-38F compliant key wrapping using AES with a 256 bit key size and iOS Boxer uses the platform to perform key wrapping using AES with a 256 bit key size. Additionally, section 8.1.12 (FCS_KYC_EXT.1) of the TSS provide a high level description of the key chaining process and identifies non-volatile storage locations that were compared to Table 17 and 18 and were found consistent. . This activity is considered satisfied.

[EC_EP] FCS_CKM_EXT.4.1 – TD0352 – *"If the platform provides the key destruction, then the evaluator shall examine the TSS to verify that it describes how the key destruction functionality is invoked.*

If the application invokes key destruction, the evaluator shall check to ensure the TSS describes each of the secret keys (keys used for symmetric encryption and/or data authentication), 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 a drive 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 'destruction of reference' (for volatile memory) is selected then the relevant interface definition is examined by the evaluator to ensure that the interface supports the selection and description in the TSS."

This activity is considered satisfied as the TSS describes in section 8.1.4 how the key destruction functionality is invoked by the platform and implemented by the application. Key destruction functionality is invoked by the platform depending on the key and its storage location. Tables 17 and 18 of the ST state whether each key destruction operation is handled by the platform or by the application. The TSS also states that the TOE either implements or invokes the platform to perform the key destruction with a single overwrite consisting of zeroes and before releasing the memory space.

Key destruction performed by the platform is done in accordance with:

> [Android] The specific cryptographic implementation for the Android platform can be found in the Samsung Electronics Co., Ltd. Samsung Galaxy Devices on Android 13 – Spring Security Target documentation (VID11342).
>
> [iOS] The specific cryptographic implementation for the iOS platform can be found in the Apple iOS 16 Security Target documentation (VID11349).

[iPadOS] The specific cryptographic implementation for the iPadOS platform can be found in the Apple iPadOS 16 Security Target documentation (VID11350).

The ST does not claim destruction of reference.

[EC_EP] FCS_CKM_EXT.5.1 – **TD0266** – *"There are two aspects of this component that require evaluation: passwords/passphrases of the length specified in the requirement (at least 64 characters) are supported, and that the characters that are input are subject to the selected conditioning function. These activities are separately addressed in the text below.*

Support for Password/Passphrase length: The evaluator shall check to ensure that the TSS describes the allowable ranges for password/passphrase lengths, and that at least 64 characters may be specified by the user.

Support for PBKDF: The evaluator shall examine the password hierarchy TSS to ensure that the formation of all keys is described and that the key sizes match that described by the ST author.

The evaluator shall check that the TSS describes the method by which the password/passphrase is first encoded and then fed to the SHA algorithm. The settings for the algorithm (padding, blocking, etc.) shall be described, and the evaluator shall verify that these are supported by the selections in this component as well as the selections concerning the hash function itself. The evaluator shall verify that the TSS contains a description of how the output of the hash function is used to form the submask that will be input into the function and is the same length as the KEK as specified in FCS_CKM_EXT.4.

For the NIST SP 800-132-based conditioning of the password/passphrase, the required assurance activities will be performed when doing the assurance activities for the appropriate requirements (FCS_COP.1.1(4) from the [AppPP]). If any manipulation of the key is performed in forming the submask that will be used to form the FEK or KEK, that process shall be described in the TSS.

No explicit testing of the formation of the submask from the input password is required.

Conditioning: No explicit testing of the formation of the authorization factor from the input password/passphrase is required. Iteration count: The evaluator shall verify that the iteration count for PBKDFs performed by the TOE comply with NIST SP 800-132 by ensuring that the TSS contains a description of the estimated time required to derive key material from passwords and how the TOE increases the computation time for password-based key derivation (including but not limited to increasing the iteration count)."

This activity is considered satisfied as the TSS states in section 8.1.5 that passwords are not accepted if they are less than the required length configurable by the administrator and greater than 512 characters. With respect to PBKDF, the TSS states that the TOE performs a password-based key derivation function in accordance with the HMAC-SHA-256 algorithm with the password string without padding, a salt of 256 bits, 10K iterations (for iOS/iPadOS) and 20K iterations (for Android) with an output cryptographic key size of 256 bits that meet NIST SP 800-132.

FCS_COP.1.1/SKC – This SFR does not contain any APP_PP or EC_EP TSS Assurance Activities.

FCS_COP.1.1/Hash – TD0717 *"The evaluator shall check that the association of the hash function with other application cryptographic functions (for example, the digital signature verification function) is documented in the TSS."*

This activity is considered satisfied as the TSS states in section 8.1.7 that the SHA-256, SHA-384, and SHA-512 hash functions are implemented by the TOE when performing cryptographic hashing in support of S/MIME functionality as defined in FCS_SMIME_EXT.1.3.

FCS_COP.1.1/Sig – TD0717 This SFR does not contain any APP_PP or EC_EP TSS Assurance Activities.

FCS_COP.1.1/KeyedHash – TD0717 This SFR does not contain any APP_PP or EC_EP TSS Assurance Activities.

[EC_EP] FCS_COP_EXT.2.1(Android), FCS_COP_EXT.2.1(iOS) & FCS_COP_EXT.2.1(iPadOS) – *"The evaluator shall examine the TSS to ensure there is a high-level description of how the key is protected and meets the appropriate specification."*

This activity is considered satisfied as the TSS states in section 8.1.10 that:

[Android] The TOE implements functionality to perform Key Wrapping using AES Key Wrap with a cryptographic key size 256-bits that meets NIST SP 800-38F for Key Wrap (section 6.2).
[iOS/iPadOS] The TOE uses the platform to perform the Key Wrapping using AES Key Wrap with a cryptographic key size 256-bits that meets NIST SP 800-38F for Key Wrap (section 6.2).

[EC_EP] FCS_IVG_EXT.1.1 – "*The evaluator shall ensure the TSS describes how IVs and tweaks are handled (based on the AES mode). The evaluator shall confirm that the IVs and tweaks meet the stated requirements."*

This activity is considered satisfied as the TSS states in section 8.1.11 that the TOE creates IVs using the CBC encryption mode, meaning that the IVs are non-repeating. This is consistent with the requirement which states, "the email client shall create IVs in the following manner: [CBC: IVs shall be non-repeating]." This is consistent with FCS_SMIME_EXT.1.2 which only claims CBC.

[EC_EP] FCS_KYC_EXT.1.1 – *"The evaluator shall verify the TSS* describes a high level description of the key hierarchy for all authorizations methods that are used to protect the encryption keys. The evaluator shall examine the TSS to ensure it describes the key chain in detail. The description of the key chain shall be reviewed to ensure it maintains a chain of keys using key wrap that meet FCS_COP_EXT.2. The evaluator shall verify the TSS* to ensure that it describes how the key chain process functions, such that it does not expose any material that might compromise any key in the chain. A high-level description should include a diagram illustrating the key hierarchy implemented and detail where all keys and keying material is stored or what it is derived from. The evaluator shall examine the key hierarchy to ensure that at no point the chain could be broken without a cryptographic exhaust or knowledge of the key within the chain and the effective strength of the data encryption key is maintained throughout the Key Chain.*

**If necessary, this information could be contained in a proprietary document and not appear in the TSS.*

If the platform provides the IV generation, then the evaluator shall examine the TSS to verify that it describes how the IV generation is invoked."

This activity is considered satisfied as the TSS specifies in section 8.1.12 a high-level description of the key hierarchy for all authorization methods (i.e. when the user is required to enter the password to unlock the application or in the case of iOS/iPadOS, an optional biometric Touch ID may be used to authenticate) that are used to protect the encryption keys. The key derived from the user entered password or biometric is processed as prescribed by FCS_COP_EXT.2 and results in a 256-bit key (Password Key).

The TSS outlines the key chaining process with an ordered bulleted list, outlining the process from when the user inputs the password or biometric unlock to finally granting access to the Boxer DB.

Additionally, a proprietary key hierarchy diagram for both Android and iOS/iPadOS was provided by the vendor to the laboratory for review. The diagram sufficiently illustrates the key hierarchy implemented and details where all keys and keying material is stored and, when applicable, where they are derived.

FCS_RBG_EXT.1.1 – *"If use no DRBG functionality is selected, the evaluator shall inspect the application and its developer documentation and verify that the application needs no random bit generation services.*

If implement DRBG functionality is selected, the evaluator shall ensure that additional FCS_RBG_EXT.2 elements are included in the ST.

If invoke platform-provided DRBG functionality is selected, the evaluator performs the following activities. The evaluator shall examine the TSS to confirm that it identifies all functions (as described by the SFRs included in the ST) that obtain random numbers from the platform RBG. The evaluator shall determine that for each of these functions, the TSS states which platform interface (API) is used to obtain the random numbers. The evaluator shall confirm that each of these interfaces corresponds to the acceptable interfaces listed for each platform below.

It should be noted that there is no expectation that the evaluators attempt to confirm that the APIs are being used correctly for the functions identified in the TSS; the activity is to list the used APIs and then do an existence check via decompilation."

This activity is considered satisfied as the TSS section 8.1.13 covers both FCS_RBG_EXT.1 and FCS_RBG_EXT.2 for Android Boxer and FCS_RBG_EXT.1 for iOS/iPadOS Boxer.

The TSS states for Android, the TOE relies on the platform's BoringSSL library to provide NIST SP 800-90A compliant AES_CTR DRBG functionality for trusted communications between the TOE and email Exchange server. It also states that the TOE implements OpenSSL to provide NIST SP 800-90A compliant AES_CTR DRBG services for the cryptographic functionality specified in the [EC_EP]. The platform API used to obtain random numbers is /dev/random and /dev/urandom depending on function (BoringSSL) and /dev/random and /dev/urandom for OpenSSL implementation depending on function. Section 6.3.1.17 identifies the FCS_RBG_EXT.2 SFR as being included in the ST and is applicable to only the Android Boxer.

The TSS states in section 8.1.13 that for iOS/iPadOS, the TOE invokes platform's CoreCrypto Module to provide NIST SP 800-90A compliant AES_CTR DRBG functionality for trusted communications between the TOE and email Exchange server. It also states that the CoreCrypto AES_CTR DRBG services also provide support for the cryptographic functionality specified in the [EC_EP]. The platform API used to obtain random numbers is SecRandomCopyBytes().

FCS_RBG_EXT.2.1 – This SFR does not contain any APP_PP or EC_EP TSS Assurance Activities.

FCS_RBG_EXT.2.2 – *"Documentation shall be produced - and the evaluator shall perform the activities - in accordance with Appendix D - Entropy Documentation and Assessment and the Clarification to the Entropy Documentation and Assessment Annex."*

This activity is considered satisfied as the Entropy Assessment Report was submitted and approved as part of the check-in process.

[EC_EP] FCS_SMIME_EXT.1 – *"The evaluator shall verify that the version of S/MIME implemented by the email client is present in the TSS. The evaluator shall also verify that the algorithms supported are specified, and that the algorithms specified are those listed for this component.*

The evaluator shall verify that the TSS describes the ContentEncryptionAlgorithmIdentifier and whether the required behavior is performed by default or may be configured.

The evaluator shall verify that the TSS describes the digestAlgorithm and whether the required behavior is performed by default or may be configured.

The evaluator shall verify that the TSS describes the signatureAlgorithm and whether the required behavior is performed by default or may be configured.

The evaluator shall verify that the TSS describes the retrieval mechanisms for both certificates and certificate revocation as well as the frequency at which these mechanisms are implemented."

This activity is considered satisfied as the TSS specifies in section 8.1.14 that the TOE implements both a sending and receiving S/MIME v4.0 Agent as defined in RFC 8551, using CMS as defined in RFCs 5652, 5754, and 3565. The TOE uses AES-256-CBC (default) and may be configured to support AES-128-CBC for the ContentEncryptionAlgorithmIdentifier. The TOE implements id-sha256 by default for the digestAlgorithm and is also configurable to use id-sha384 and id-sha512. The TOE uses sha256withRSAEncryption for the signatureAlgorithm by default. The algorithms described in the TSS are consistent with those specified in the SFR.

The TSS also states that certificate revocation information is received from the OCSP responder at a frequency defined by the server, but may be overridden by a UEM administrator to a configured value via UEM. The TSS goes on to say that certificates are pulled from signed email when received in the application in order to validate the authenticity and integrity of the email. The UEM is a Mobile Device Management (MDM) product.

FCS_STO_EXT.1.1(1) & (2) – *"The evaluator shall check the TSS to ensure that it lists all persistent credentials (secret keys, PKI private keys, or passwords) needed to meet the requirements in the ST. For each of these items, the evaluator shall confirm that the TSS lists for what purpose it is used, and how it is stored."*

This activity is considered satisfied as the TSS provides in section 8.1.15 a series of tables for iOS/iPadOS and Android listing all persistent (non-volatile) and non-persistent (volatile) credentials, which is consistent with the SFR requirement. For each item listed in the table, the TSS lists the purpose in the "Description" column and how it is stored under the "Non-volatile" column. Boxer on Android platforms implements and invokes the OS to perform this function. The table clearly identifies which is responsible. Boxer on iOS/iPadOS invokes the OS to implement this function.

The TSS states in section 8.1.16 that the Boxer application is designed to store certification revocation status information, including the certificate identifier, revocation status, validation failure reason, next revocation check date, and last revocation checked date. This information is overwritten when information is refreshed. The revocation status information is deleted upon the Boxer application removal. Boxer on Android platforms implements this function. Boxer on iOS/iPadOS invokes the OS to implement this function.

FDP_DAR_EXT.1 – *"The evaluator shall examine the TSS to ensure that it describes the sensitive data processed by the application. The evaluator shall then ensure that the following activities cover all of the sensitive data identified in the TSS.*

If not store any sensitive data is selected, the evaluator shall inspect the TSS to ensure that it describes how sensitive data cannot be written to non-volatile memory. The evaluator shall also ensure that this is consistent with the filesystem test below.

***For Android:** The evaluator shall inspect the TSS and verify that it describes how files containing sensitive data are stored with the MODE_PRIVATE flag set.*

***For iOS:** The evaluator shall inspect the TSS and ensure that it describes how the application uses the Complete Protection, Protected Unless Open, or Protected Until First User Authentication Data Protection Class for each data file stored locally."*

This activity is considered satisfied as the TSS describes in section 8.2.1 the sensitive data processed by the application for Android and iOS/iPadOS. For Android the TSS states that sensitive data includes: calendar,

address book (i.e. contacts), system accounts, and profile. For iOS/iPadOS, sensitive data includes: calendar, address book, system accounts, and profile.

For Android, the TSS also states that "[t]he TOE's file creation scheme requires sensitive data files to be saved with the MODE_PRIVATE flag set. All instances where files containing sensitive data are stored call the getSharedPreferences(String name, int mode) method (which is an overridden method defined in the Boxer application source code) with the "MODE_PRIVATE" flag as the second parameter."

For iOS/iPadOS, the TSS also states that "[t]he TOE's file creation scheme requires sensitive data files to use Protected Until First User Authentication Data Protection Class for each data file stored locally." "All instances where sensitive data files are stored rely on the NSFileProtectionCompleteUntilFirstUserAuthentication declaration in "VMwareBoxer.entitlements", which is contained in the TOE .ipa file and is enforced in code.

FDP_DEC_EXT.1.1(Android), (iOS) & (IPadOS) – This SFR does not contain any APP_PP or EC_EP TSS Assurance Activities.

FDP_DEC_EXT.1.2(Android), (iOS) & (IPadOS) – This SFR does not contain any APP_PP or EC_EP TSS Assurance Activities.

FDP_NET_EXT.1.1 – This SFR does not contain any APP_PP or EC_EP TSS Assurance Activities.

[EC_EP] FDP_NOT_EXT.1.1 – *"The evaluator shall ensure that the TSS describes notifications of S/MIME status, including whether S/MIME status is also indicated upon viewing a list of emails."*

This activity is considered satisfied as the TSS describes in section 8.2.4 the format for S/MIME notifications and the location of the notification icon when viewing an email. Additionally, the TSS states that the S/MIME status is indicated upon viewing a list of emails, but the signature validity is not displayed until the message is opened.

[EC_EP] FDP_SMIME_EXT.1.1 – *"The evaluator shall verify that the TSS contains a description of the S/MIME implementation and its use to protect mail from undetected modification using digital signatures and unauthorized disclosure using encryption. The evaluator shall verify that the TSS describes whether signature verification and decryption occur at receipt or viewing of the message contents, and whether messages are stored with their S/MIME envelopes."*

This activity is considered satisfied as the TSS states in section 8.2.5 that the TOE uses S/MIME for signing, encrypting, verifying, and decrypting email and is implemented as specified in FCS_SMIME_EXT.1. The TSS states that the signature verification and decryption occur at the receipt of the message. The TSS also states that messages are not stored with their S/MIME envelopes.

[APP_PP] FIA_X509_EXT.1.1 – *"The evaluator shall ensure the TSS describes where the check of validity of the certificates takes place. The evaluator ensures the TSS also provides a description of the certificate path validation algorithm."*

This activity is considered satisfied as the TSS states in section 8.3.1 that the TOE application, regardless of platform, performs certificate validation for certificates used for TLS communications. It also describes the certificate path validation algorithm.

FIA_X509_EXT.1.2 – This SFR does not contain any APP_PP or EC_EP TSS Assurance Activities.

[APP_PP] FIA_X509_EXT.2.2 – *"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. The evaluator shall verify that any distinctions between trusted channels are described."

This activity is considered satisfied as the TSS states in section 8.3.2 that the use of certificates is enabled by default for TLS authentication to the Exchange server. The administrator may also specify the path to an OCSP responder so that revocation status can be checked during authentication. The trusted CA certificates to be used by the TOE are specified through the UEM console. The UEM administrator is able to specify the default action when the application/platform cannot reach the OCSP responder, so that the TOE will either reject the certificate or accept the certificate, depending on the configuration and status.

[EC_EP] FIA_X509_EXT.3 – *"The evaluator shall check the TSS to ensure that it describes how the email client chooses which certificates to use so that the email client can use the certificates.*

The evaluator shall examine the TSS to confirm that it describes the behavior of the email client when a connection cannot be established during the validity check of a certificate used in establishing a trusted channel and protecting email."

This activity is considered satisfied as the TSS states in section 8.3.3 that certificates used for S/MIME functionality are transmitted from the UEM server to the Boxer application upon initial launch of the application and subsequent launches if new certificates are available.

The TSS also states that when the TOE cannot establish a connection to the OCSP responder during the certificate validity check, the application can be configured by an UEM administrator via UEM to either:

- Reject the certificate.
- Accept the certificate if the last revocation status is valid. Reject the certificate if the last known revocation status is unknown or was revoked.

FMT_CFG_EXT.1.1 – *"The evaluator shall check the TSS to determine if the application requires any type of credentials and if the application installs with default credentials."*

The TSS states in section 8.4.1 that there are no default credentials for the TOE. All credentials would be pre-existing on the Exchange server or UEM server, which are separate entities to the TOE.

FMT_CFG_EXT.1.2 – This SFR does not contain any APP_PP or EC_EP TSS Assurance Activities.

FMT_MEC_EXT.1.1 – *"The evaluator shall review the TSS to identify the application's configuration options (e.g. settings) and determine whether these are stored and set using the mechanisms supported by the platform or implemented by the application in accordance with the PP-Module for File Encryption. At a minimum the TSS shall list settings related to any SFRs and any settings that are mandated in the operational guidance in response to an SFR.*

Conditional: If "implement functionality to encrypt and store configuration options as defined by FDP_PRT_EXT.1 in the PP-Module for File Encryption" is selected, the evaluator shall ensure that the TSS identifies those options, as well as indicates where the encrypted representation of these options is stored."

This activity is considered satisfied as the TSS states in section 8.4.2 that for both Android and iOS/iPadOS, the only user configurable sensitive TSF datum is the application password as defined in FMT_MOF_EXT.1 and are stored using the appropriate platform mechanism.

The evaluator shall inspect the TSS and verify that it describes what Android API is used (and provides a link to the documentation of the API) when storing configuration data.

This activity is considered satisfied as the TSS states in section 8.4.2 that the SharedPreferences XML file is the Android API and a link to the documentation has been provided.

[EC_EP] FMT_MOF_EXT.1.1 – *"The evaluator shall verify that the TSS describes those management functions which may only be configured by the email client platform administrator and cannot be over-ridden by the user when set according to policy.*

Change Password: The evaluator shall examine the Operational Guidance to ensure that it describes how the password/passphrase-based authorization factor is to be changed.

Disable Key Recovery: If the email client supports key recovery, this must be stated in the TSS. The TSS shall also describe how to disable this functionality. This includes a description of how the recovery material is provided to the recovery holder.

Cryptographic Configuration: The evaluator shall determine from the TSS for other requirements (FCS_*) what portions of the cryptographic functionality are configurable."

This activity is considered satisfied as the TSS describes in section 8.4.3 the management functions which may only be configured by the UEM administrator and cannot be overridden by the user. The UEM administrator can configure the password length, cryptographic functionality such as specifying what key sizes are used for the cryptographic algorithms in FCS_SMIME_EXT.1, S/MIME cryptographic assignments, OCSP retrieval frequency, and enabling/disabling the plaintext only mode globally. The TSS states that the mobile device user is not capable of changing these settings from the mobile device. This list is consistent with the selections and assignments in the SFR.

The TSS provides a reference to the Supplemental Administrative Guidance document (AGD) on how the user defined password authorization factor can be changed.

The email client does not support key recovery.

FMT_SMF.1.1 – This SFR does not contain any APP_PP or EC_EP TSS Assurance Activities.

FPR_ANO_EXT.1.1 – *"The evaluator shall inspect the TSS documentation to identify functionality in the application where PII can be transmitted."*

This activity is considered satisfied as the TSS states in section 8.5.1 that the TOE application does not collect personally identifiable information (PII) for administrators or users. Therefore, the TOE application will not transmit PII data over the network unless the user of the mobile device includes such information in the free text email. The free text in an email is outside the TOE's scope of control.

FPT_AEX_EXT.1.1 – **TD0798** *"The evaluator shall ensure that the TSS describes the compiler flags used to enable ASLR when the application is compiled." If any explicitly-mapped exceptions are claimed, the evaluator shall check that the TSS identifies these exceptions, describes the static memory mapping that is used, and provides justification for why static memory mapping is appropriate in this case.*

This activity is considered satisfied as the TSS states in section 8.6.1 that for iOS/iPadOS, the TOE is compiled using the LD_NO_PIE=NO compilation flag to ensure it is a Position Independent Executable (ASLR) and for Android, the TOE is compiled using the -fPIE and -pie compilation flags to ensure it is a Position Independent Executable (ASLR). Additionally, this section states that uses of mmap have the explicit memory address location parameter set to NULL (or 0) with the exception of when mmap is called to reallocate memory to expand the Boxer database file map.

FPT_AEX_EXT.1.2 – This SFR does not contain any APP_PP or EC_EP TSS Assurance Activities.

FPT_AEX_EXT.1.3 – This SFR does not contain any APP_PP or EC_EP TSS Assurance Activities.

FPT_AEX_EXT.1.4 – This SFR does not contain any APP_PP or EC_EP TSS Assurance Activities.

FPT_AEX_EXT.1.5 – **TD0815** *(Conditional: The PE or ELF automated tests fail) The evaluator shall ensure that the TSS describes the stack-based buffer overflow compiler flags.*

This activity is considered satisfied as the TSS states in Section 8.6.1 that the TOE was compiled using the-fstack-protector-all compilation flag for all platforms.

[EC_EP] FPT_AON_EXT.1.1 – *"The evaluator shall verify that the TSS describes whether the email client is capable of loading trusted add-ons."*

This activity is considered satisfied as the TSS states in section 8.6.2 that the TOE does not support the installation of trusted or untrusted add-ons.

FPT_API_EXT.1.1 – *"The evaluator shall verify that the TSS lists the platform APIs used in the application."*

This activity is considered satisfied as the TSS lists in section 8.6.3 separately for Android and iOS/iPadOS the platform APIs used in the TOE application. For each platform API referenced in the TSS, a search was performed to locate the corresponding platform API documentation and verified that it was supported for the platform OS version.

FPT_IDV_EXT.1.1 – *"If "other version information" is selected the evaluator shall verify that the TSS contains an explaination of the versioning methodology."*

This activity is considered satisfied as the TSS states in section 8.6.4 that the TOE version format is for Android is "YY.MM.PP.BB". The vendor operates on a monthly release cycle to incorporate updates and fixes. The first number is based on the last two digits of the year of the release date and the second number is the 2-digit numerical representation of the month of the release date, the third number is based on the patch release under the monthly release number, and the fourth number is based on the internal build number that has been released to the Google Play store.

The TOE version format for iOS/iPadOS is "YY.MM.PP". The first number is based on the last two digits of the year of the release date. The second number is the 2-digit numerical representation of the month of the release date, and the third number is based on the patch release under the monthly release number.

FPT_LIB_EXT.1.1 – This SFR does not contain any APP_PP or EC_EP TSS Assurance Activities.

FPT_TUD_EXT.1.1 – This SFR does not contain any APP_PP or EC_EP TSS Assurance Activities.

FPT_TUD_EXT.1.2 – This SFR does not contain any APP_PP or EC_EP TSS Assurance Activities.

FPT_TUD_EXT.1.3 – This SFR does not contain any APP_PP or EC_EP TSS Assurance Activities.

FPT_TUD_EXT.1.4 – *"The evaluator shall verify that the TSS identifies how updates to the application are signed by an authorized source. The definition of an authorized source must be contained in the TSS. The evaluator shall also ensure that the TSS (or the operational guidance) describes how candidate updates are obtained."*

This activity is considered satisfied as the TSS states in section 8.6.6 that prior to the installation of the TOE, the Boxer software is digitally signed using a Verisign X.509v3 certificate. The software is verified by the UEM server prior to being pushed to the mobile device UEM agent for the initial installation.

The TSS also states that updates to the TOE are provided by the Google Play Store (for Android) or the Apple Store (for iOS/iPadOS) over HTTPS/TLS. Once the update has been completed by the developer, it is then digitally signed by the developer, sent to the Google Play Store/Apple Store. The Google

Play/Apple Store will then verify the signature and will sign the update with its own signature. When the update gets sent to the mobile device, the mobile device will verify the signature from the Google Play Store/Apple Store.

FPT_TUD_EXT.1.5 – *"The evaluator shall verify that the TSS identifies how the application is distributed. If "with the platform" is selected the evaluated shall perform a clean installation or factory reset to confirm that TOE software is included as part of the platform OS. If "as an additional package" is selected the evaluator shall perform the tests in FPT_TUD_EXT.2."*

This activity is considered satisfied as the SFR selection states the application is distributed as an additional software package to the platform OS. The TSS states in section 8.6.6 that the application for Android is packaged in .apk format and for iOS/iPadOS it is packaged in .ipa format. Additionally, the evaluator performed the tests as prescribed in FPT_TUD_EXT.2.

FPT_TUD_EXT.2.1 – This SFR does not contain any APP_PP or EC_EP TSS Assurance Activities.

FPT_TUD_EXT.2.2 – This SFR does not contain any APP_PP or EC_EP TSS Assurance Activities.

FPT_TUD_EXT.2.3 – "The evaluator shall verify that the TSS identifies how the application installation package is signed by an authorized source. The definition of an authorized source must be contained in the TSS."

This activity is considered satisfied as the TSS states in section 8.6.6 that the TOE and updates to the TOE are provided by the Google Play Store (Android) or Apple App Store (iOS) over HTTPS/TLS. Once the update has been completed by the developer, it is then digitally signed by the developer and sent to the Google Play Store/Apple App Store. The TOE software is digitally signed using a Verisign X.509v3 certificate. The Google Play Store/Apple App Store will then verify the signature and will sign the update with its own signature. When the update gets sent to the mobile device, the mobile device will verify the signature from the Google Play Store/Apple App Store.

FTP_DIT_EXT.1.1(Android), (iOS) & (IPadOS) – *"For platform-provided functionality, the evaluator shall verify the TSS contains the calls to the platform that TOE is leveraging to invoke the functionality."*

This activity is considered satisfied as the TSS states in section 8.7.1 identifies the calls to the platform that the TOE leverages to invoke the secure communication functionality.

For Android: The TOE invokes its host platform to establish a TLSv1.2 channel with the Exchange Server platform to secure all transmitted data via *javax.net.ssl.SSLSocketFactory*. Once the TLS connection is established between the platforms, the TOE invokes its host platform to use ActiveSync for exchanging email with the Exchange Server via the *ApacheHttpClient* library which internally uses *java.net.HttpURLConnection*. In this instance, the TOE platform acts as the TLS client to initiate the secure communications to the Exchange server to send and receive emails.

For iOS and iPadOS: The TOE invokes its host platform to establish a TLSv1.2 channel with the Exchange Server platform to secure all transmitted data via the *NSURLCredential.credentialForTrust* system method. Once the TLS connection is established between the platforms, the TOE invokes its host platform to use ActiveSync for exchanging email with the Exchange Server via the *NSURLSessionTask*. In this instance, the TOE platform acts as the TLS client to initiate the secure communications to the Exchange server to send and receive emails.

[EC_EP] FTP_ITC_EXT.1 – *"The evaluator shall examine the TSS to determine that it describes the details of the email client connecting to a Mail Transfer Agent in terms of the trusted connection (i.e., TLS) according to FTP_DIT_EXT.1 [AppPP], along with email client-specific options or procedures that might not be reflected in the specification."*

This activity is considered satisfied as the TSS states in section 8.7.2 that the TOE invokes the platform to communicate with an Exchange server for its primary function as an email client. The TOE initiates the actual ActiveSync protocol layer communications after it invokes the OS to establish the trusted channel that ActiveSync requires.

3 Operational Guidance Assurance Activities

The evaluation team completed the testing of the Operational Guidance, which includes the review of the *VMware Workspace ONE Boxer Email Client Version 23.11 Supplemental Administrative Guidance for Common Criteria v1.0* (AGD) document and confirmed that the Operational Guidance contains all Assurance Activities as specified by the *Protection Profile for Application Software Version 1.4 [APP_PP]* and *Application Software Extended Package for Email Clients v2.0 [EC_EP]*. The evaluators reviewed the APP_PP and EC_EP to identify the security functionality that must be discussed for the operational guidance. This is prescribed by the Assurance Activities for each SFR and the AGD SARs. The evaluators have listed below each of the SFRs defined in the APP_PP and EC_EP that have been claimed by the TOE (some SFRs are conditional or optional) as well as the AGD SAR, along with a discussion of where in the operational guidance the associated Assurance Activities material can be found. The AGD includes references to other guidance documents that must be used to properly install, configure, and operate the TOE in its evaluated configuration. The AGD and its references to other VMware Workspace ONE Boxer Email Client Version 23.11 guidance documents were reviewed to assess the Operational Guidance Assurance Activities. The AGD contains references to these documents in Chapter 4 and these references can also be found below:

The following references are used in this section of the document: Customize list. Should be consistent with other document declarations.

[1] VMware Workspace ONE Boxer Admin Guide
[2] VMware Workspace ONE Boxer for Android User Guide
[3] VMware Workspace ONE Boxer for iOS User Guide
[4] VMware Workspace ONE Boxer Email Client Version 23.11 Security Target v1.0 (ST)

FCS_CKM_EXT.1.1 – TD0717 This SFR does not contain any APP_PP or EC_EP AGD Assurance Activities.

FCS_CKM.1.1/AK – TD0717 *"The evaluator shall verify that the AGD guidance instructs the administrator how to configure the TOE to use the selected key generation scheme(s) and key size(s) for all uses defined in this PP."*

Section 6.2.1 of the AGD states that the TOE requires no special configuration in order to use the selected key generation schemes, as they are provided by the TOE platform.

FCS_CKM.2.1 – *"The evaluator shall verify that the AGD guidance instructs the administrator how to configure the TOE to use the selected key establishment scheme(s)."*

Section 6.2.1 of the AGD states that the TOE requires no special configuration in order to use the selected key establishment schemes, as they are provided by the TOE platform.

[EC_EP] FCS_CKM_EXT.3.1 – This SFR does not contain any APP_PP or EC_EP AGD Assurance Activities.

[EC_EP] FCS_CKM_EXT.4.1 – This SFR does not contain any APP_PP or EC_EP AGD Assurance Activities.

[EC_EP] FCS_CKM_EXT.5.1 – **TD0266** –*"The evaluator shall check the Operational Guidance to determine that there are instructions on how to generate large passwords/passphrases, and instructions on how to configure the password/passphrase length (and optional complexity settings) to provide entropy*

commensurate with the keys that the authorization factor is protecting. This is important because many default settings for passwords/passphrases will not meet the necessary entropy needed as specified in this EP."

The AGD states in Section 7.2.1 that TOE users are responsible for setting and changing their password/passphrase by following the steps described outlined in the same section. There is a "Note to the End User" to create a strong passcode of 8 or more characters (up to 512 characters) and use each of the 4 character sets because this passcode is used to derive cryptographic keys to encrypt the Boxer database.

Additionally, there is a "Note to the UEM Administrator" includes additional information that the passcode is used to derive the 256 bit cryptographic key to encrypt the Boxer database. It also states the Boxer is designed to compensate for a bad passcode by ensuring that there is enough entropy to create the key, but still highly recommends the setting of the minimum length passcode to 8 or more characters. There is an additional pointer to other passcode complexity parameters that were not required to be tested as part of this evaluation.

The section goes on to describe how the declared password composition requirements are configured via the UEM Console.

FCS_COP.1.1/SKC – TD0717 *"The evaluator checks the AGD documents to determine that any configuration that is required to be done to configure the functionality for the required modes and key sizes is present."*

The AGD states in section 6.2.2.1 how to configure the TOE to use the required encryption algorithms and key sizes for S/MIME encryption and decryption.

FCS_COP.1.1/Hash – TD0717 This SFR does not contain any APP_PP or EC_EP AGD Assurance Activities.

FCS_COP.1.1/Sig – TD0717 This SFR does not contain any APP_PP or EC_EP AGD Assurance Activities.

FCS_COP.1.1/KeyedHash) – TD0717 – This SFR does not contain any APP_PP or EC_EP AGD Assurance Activities.

[EC_EP] FCS_COP_EXT.2.1(Android), FCS_COP_EXT.2.1(iOS) & FCS_COP_EXT.2.1(iPadOS) – This SFR does not contain any APP_PP or EC_EP AGD Assurance Activities.

[EC_EP] FCS_IVG_EXT.1.1 – This SFR does not contain any APP_PP or EC_EP AGD Assurance Activities.

[EC_EP] FCS_KYC_EXT.1.1 – This SFR does not contain any APP_PP or EC_EP AGD Assurance Activities.

FCS_RBG_EXT.1.1 – This SFR does not contain any APP_PP or EC_EP AGD Assurance Activities.

FCS_RBG_EXT.2.1 – This SFR does not contain any APP_PP or EC_EP AGD Assurance Activities.

FCS_RBG_EXT.2.2 – This SFR does not contain any APP_PP or EC_EP AGD Assurance Activities.

[EC_EP] FCS_SMIME_EXT.1 – *"The evaluator shall also review the Operational Guidance to ensure that it contains instructions on configuring the email client such that it complies with the description in the TSS.*

If the TSS indicates that the algorithms in FCS_SMIME_EXT.1.2 must be configured to meet the requirement, the evaluator shall verify that the AGD guidance includes the configuration of this ID.

If the TSS indicates that the algorithms in FCS_SMIME_EXT.1.3 must be configured to meet the requirement, the evaluator shall verify that the AGD guidance includes the configuration.

If the TSS indicates that the algorithms in FCS_SMIME_EXT.1.4 must be configured to meet the requirement, the evaluator shall verify that the AGD guidance includes the configuration of this ID.

If the TSS indicates that the mechanisms in FCS_SMIME_EXT.1.7 are configurable, the evaluator shall verify that the AGD guidance includes the configuration of these mechanisms."

The AGD specifies in section 6.2.2.1 the steps required to enable the usage of the AES-128-CBC and AES-256-CBC encryption algorithms for S/MIME encryption and decryption.

The AGD specifies in section 6.2.2.2 the steps required to enable the usage of the id-sha256, id-sha384, id-sha512 message digest algorithms for S/MIME signing.

The AGD specifies in section 6.2.2.3 that no configuration is needed to enable the sha256withRSAEncryption signature algorithm because it is the only enabled algorithm by default.

The AGD specifies in section 6.3.1 the configuration steps, parameters, and values for setting the certificate revocation check frequency behavior.

FCS_STO_EXT.1.1(1) & (2) – This SFR does not contain any APP_PP or EC_EP AGD Assurance Activities.

FDP_DAR_EXT.1 – This SFR does not contain any APP_PP or EC_EP AGD Assurance Activities.

FDP_DEC_EXT.1.1(Android), (iOS) & (IPadOS) – *"The evaluator shall perform the platform-specific actions below and inspect user documentation to determine the application's access to hardware resources. The evaluator shall ensure that this is consistent with the selections indicated. The evaluator shall review documentation provided by the application developer and for each resource which it accesses, identify the justification as to why access is required."*

Section 7.4 of the AGD lists the hardware resources that are requested by the TOE. The list of hardware resources in the AGD is consistent with the selections from the ST. For each hardware resource, the AGD provides a justification for why access is required. For example, network connectivity access is requested to sync e-mail with the Exchange server.

FDP_DEC_EXT.1.2(Android), (iOS) & (IPadOS) – *"The evaluator shall perform the platform-specific actions below and inspect user documentation to determine the application's access to sensitive information repositories. The evaluator shall ensure that this is consistent with the selections indicated. The evaluator shall review documentation provided by the application developer and for each sensitive information repository which it accesses, identify the justification as to why access is required."*

Section 7.4 of the AGD lists the sensitive information repositories that are requested by the TOE. The list of sensitive information repositories in the AGD is consistent with the selections from the ST. For each sensitive information repository, the AGD provides a justification for why access is required. For example, calendar access is requested to add or modify calendar events and send email to guests.

FDP_NET_EXT.1.1 – This SFR does not contain any APP_PP or EC_EP AGD Assurance Activities.

[EC_EP] FDP_NOT_EXT.1.1 – *"The evaluator shall verify that the AGD guidance provides a description (with appropriate visual figures) of the S/MIME status notification(s), including how each of the following are indicated: encryption, verified and validated signature, and unverified and unvalidated signature."*

The AGD in section 7.2.3 provides both a textual description and visual figure for each of the S/MIME status notifications, including: encryption verified, validated signature, encryption verified and validated signature, and unverified and unvalidated signature for both Android and iOS / iPadOS TOE platforms.

[EC_EP] FDP_SMIME_EXT.1.1 – *"The evaluator shall ensure that the AGD guidance includes instructions for configuring a certificate for S/MIME use and instructions for signing and encrypting email."*

The AGD in section 6.2.2.4 lists the instructions for configuring the TOE to obtain and use S/MIME certificates for signing and encrypting email. Section 7.2.2 describes how to sign and encrypt and email.

FIA_X509_EXT.1.1 – This SFR does not contain any APP_PP or EC_EP AGD Assurance Activities.

FIA_X509_EXT.1.2 – This SFR does not contain any APP_PP or EC_EP AGD Assurance Activities.

[APP_PP] FIA_X509_EXT.2.2 – *"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."*

Section 6.3.2 of the AGD lists the configuration procedures for setting the conditions under which the TOE will either reject or accept a presented S/MIME or TLS certificate when the OCSP responder is unavailable.

[EC_EP] FIA_X509_EXT.3 – *"The evaluator shall verify that the administrative guidance contains any necessary instructions for configuring the operating environment so that the email client can use the certificates."*

The AGD in section 6.2.2.4 lists the instructions for configuring the TOE to obtain and use S/MIME certificates for signing and encrypting email.

FMT_CFG_EXT.1.1 – This SFR does not contain any APP_PP or EC_EP AGD Assurance Activities.

FMT_CFG_EXT.1.2 – This SFR does not contain any APP_PP or EC_EP AGD Assurance Activities.

FMT_MEC_EXT.1.1 – This SFR does not contain any APP_PP or EC_EP AGD Assurance Activities.

[EC_EP] FMT_MOF_EXT.1.1 – *"The evaluator shall examine the operational guidance to verify that it includes instructions for an email client platform administrator to configure the functions listed in FMT_MOF_EXT.1.1.*

Disable Key Recovery: If the email client supports key recovery, the guidance for disabling this capability shall be described in the AGD documentation.

Cryptographic Configuration: The evaluator shall review the AGD documentation to determine that there are instructions for manipulating all of the claimed mechanisms."

Section 7.2 of the AGD provides steps for configuring whether plaintext only mode is enabled or disabled globally.

Section 6.2.2 of the AGD provides steps for configuring the set of cryptographic algorithms as well as the cryptographic functionality (e.g. algorithm key size) defined in FCS_SMIME_EXT.1 for sending and receiving S/MIME messages.

Section 7.2.1 of the AGD provides steps for the user to change the TOE password/passphrase authentication credential as well as procedures for the administrator to configure the password length complexity policy.

Section 6.3.1 of the AGD lists the steps for the administrator to configure the OCSP retrieval frequency behavior.

The AGD states in section 7.2 that the TOE does not support key recovery.

FMT_SMF.1.1 – *"The evaluator shall verify 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."*

Section 7.2.1 of the AGD provides steps for the user to change the TOE password/passphrase authentication credential. This is the only management function the owner/user of the mobile device can change.

FPR_ANO_EXT.1.1 – This SFR does not contain any APP_PP or EC_EP AGD Assurance Activities.

FPT_AEX_EXT.1.1 – This SFR does not contain any APP_PP or EC_EP AGD Assurance Activities.

FPT_AEX_EXT.1.2 – This SFR does not contain any APP_PP or EC_EP AGD Assurance Activities.

FPT_AEX_EXT.1.3 – This SFR does not contain any APP_PP or EC_EP AGD Assurance Activities.

FPT_AEX_EXT.1.4 – This SFR does not contain any APP_PP or EC_EP AGD Assurance Activities.

FPT_AEX_EXT.1.5 – This SFR does not contain any APP_PP or EC_EP AGD Assurance Activities.

[EC_EP] FPT_AON_EXT.1.1 – "*The evaluator shall examine the operational guidance to verify that it includes instructions on loading trusted add-on sources."*

The AGD states in section 7.2.4 that the TOE does not support the installation of trusted or untrusted add-ons.

FPT_API_EXT.1.1 – This SFR does not contain any APP_PP or EC_EP AGD Assurance Activities.

FPT_IDV_EXT.1.1 – This SFR does not contain any APP_PP or EC_EP AGD Assurance Activities.

FPT_LIB_EXT.1.1 – This SFR does not contain any APP_PP or EC_EP AGD Assurance Activities.

FPT_TUD_EXT.1.1 – *"The evaluator shall check to ensure the guidance includes a description of how updates are performed."*

The AGD states in section 7.3 that updates to the TOE are provided by the Google Play Store (for Android) and the Apple Store (for iOS) over HTTPS/TLS.

FPT_TUD_EXT.1.2 – *"The evaluator shall verify guidance includes a description of how to query the current version of the application."*

The AGD states in section 7.3 for both iOS and Android to navigate to the Settings→About to determine the current version.

FPT_TUD_EXT.1.3 – This SFR does not contain any APP_PP or EC_EP AGD Assurance Activities.

FPT_TUD_EXT.1.4 – This SFR does not contain any APP_PP or EC_EP AGD Assurance Activities.

FPT_TUD_EXT.1.5 – This SFR does not contain any APP_PP or EC_EP AGD Assurance Activities.

FPT_TUD_EXT.2.1 – This SFR does not contain any APP_PP or EC_EP AGD Assurance Activities.

FPT_TUD_EXT.2.2 – This SFR does not contain any APP_PP or EC_EP AGD Assurance Activities.

FTP_DIT_EXT.1.1(Android), (iOS) & (IPadOS) – This SFR does not contain any APP_PP or EC_EP AGD Assurance Activities.

[EC_EP] FTP_ITC_EXT.1 – *"The evaluator shall confirm that the operational guidance contains instructions for establishing the connection to the Mail Transfer Agent."*

Section 6.2.3 of the AGD lists the steps to be performed by the administrator to configure the TOE to communicate with the Exchange server. Section 6.5 of the AGD describes the procedures for the user to establish a connection and authenticate to the Exchange server.

4 Test Assurance Activities

4.1 Test Configuration

The evaluation team conducted testing at the Booz Allen Common Criteria Testing Laboratory (CCTL) on an isolated network. The evaluation team configured the TOE for testing according to the *VMware Workspace ONE Boxer Email Client Version 23.11 Supplemental Administrative Guidance for Common Criteria Version 1.0* (AGD) document. The evaluation team set up a test environment for the independent functional testing that allowed them to perform the assurance activities against the TOE over the SFR relevant interfaces.

The TOE was configured to communicate with the following environment components:

- E1: OCSP Responder server (Windows Server 2019 (Version 1809)).
    - Microsoft Online Certificate Status Protocol Responder
- E2: VMware Workspace ONE UEM version 20.8.0.3 (2008) server was (Windows Server 2019 (Version 1809)).
- E3: Exchange server (Windows Server 2019 (Version 1809)).
    - Exchange Server 2019 CU7; Version 15.2 (Build 721.2)
- E4: Mobile device OS platform store
- E5: Mobile Device for running the TOE software application (Android, iOS, & iPadOS)

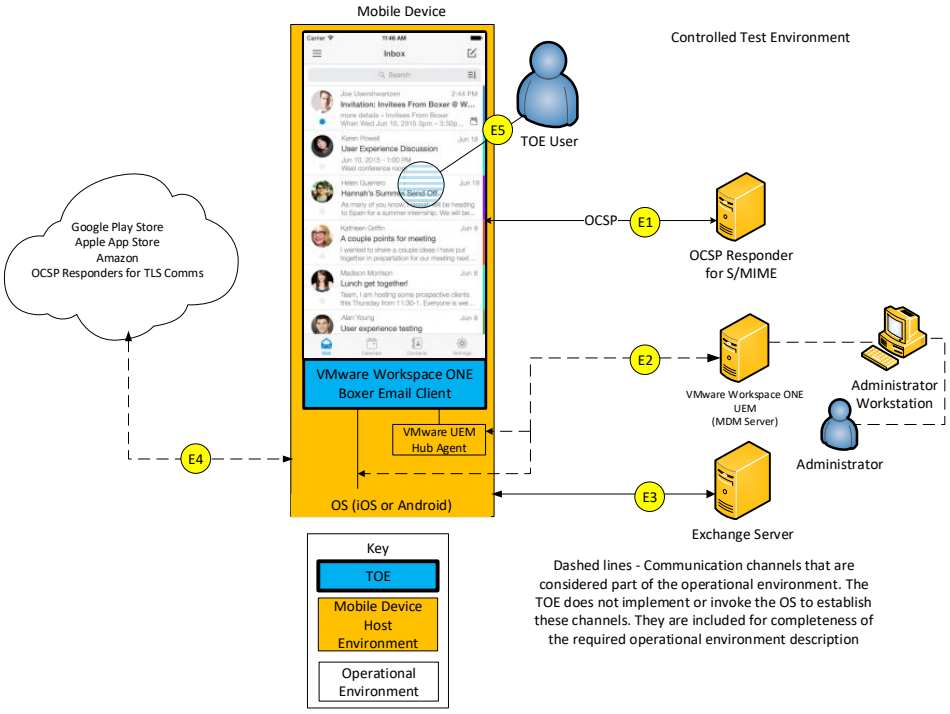

Figure 1 - VMware Workspace ONE Boxer Email Client Version 23.11 TOE Boundary

The following test tools were installed in the operational environment on multiple test workstations and servers for testing purposes:

- Android Asset Packaging Tool (aapt) 2.19-10154469
- Android Debug Bridge version 1.0.41, Version 33.0.3-8952118
- Binary Walking Tool (Binwalk v2.2.3)
- iOS Network Analysis Tool (Xcode v15.0 build 15A240d)
- Memory Dump Tool (Frida V 16.1.10)
- Man-in-the-Middle (MITM) Packet Modification Tool (ettercap 0.8.3.1)
- postfix version 3.1.15
- OpenSSL 1.1.l 10 Sep 2019
- Python version 3.11.2
- Wireshark version 4.0.5
- Xcode Version: 15.0 Build version 15A240d

The following test tools were installed on the mobile device only when needed for testing purposes:

- iOS Keychain Dump Tool (Vmware tool)
- Memory Dump Tool (Frida V 16.1.10)

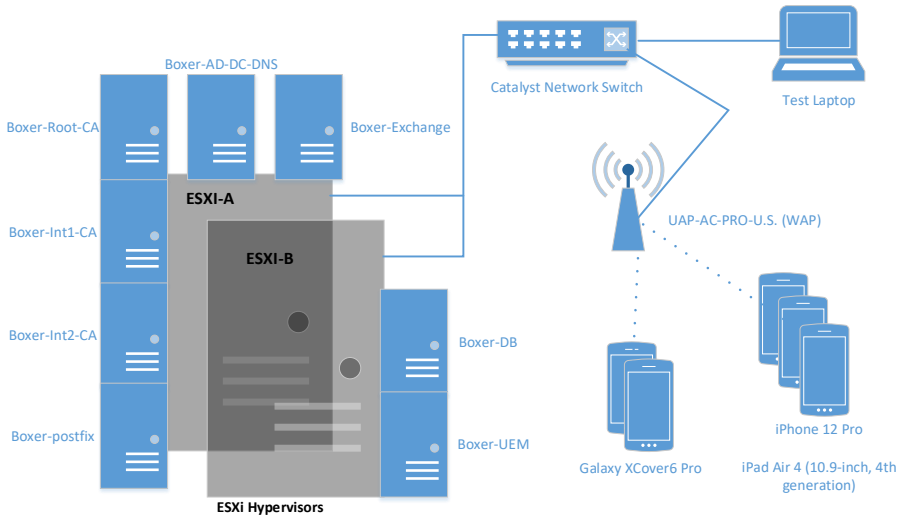

Figure 2 - Test Configuration

4.2 Omission Justification

There were no testing omissions because there was no sampling of testing, as all required test assurance activities were performed against the TOE installed on the three platforms (Android 13, iOS 16, and iPadOS 16) that were claimed in the Security Target.

4.3 Test Cases

The evaluation team completed the functional testing activities within the Booz Allen Common Criteria Testing Laboratory (CCTL), located in Laurel, MD between August 2023 - March 2024. The evaluation team conducted a set of testing that includes all ATE Assurance Activities as specified by the *Protection Profile for Application Software Version 1.4 [APP_PP]* and *Application Software Extended Package for Email Clients v2.0 [EC_EP]*. The evaluators reviewed the APP_PP and EC_EP to identify the security functionality that must be verified through functional testing. This is prescribed by the Assurance Activities for each SFR.

If an SFR is not listed, one of the following conditions applies:

- The Assurance Activity for the SFR specifically indicates that it is simultaneously satisfied by completing a test Assurance Activity for a different SFR.
- The Assurance Activity for the SFR does not specify any actions related to ATE activities (e.g. FCS_KYC_EXT.1).

Note that some SFRs do not have Assurance Activities associated with them at the element level (e.g. FCS_SSH_EXT.1.1). In such cases, testing for the SFR is considered to be satisfied by completion of all Assurance Activities at the component level.

The following lists for each ATE Assurance Activity, the test objective, test instructions, test steps, and test results. Note that unless otherwise specified, the test configuration is to be in the evaluated configuration as defined by the AGD. For example, some tests require the TOE to be brought out of the evaluated configuration to temporarily disable cryptography to prove that the context of transmitted data is accurate. As part of the cleanup for each test, the TOE is returned to the evaluated configuration.

4.3.1 Cryptographic Support

ATTENTION:
[iOS] TLS communication and S/MIME cryptographic services for Boxer application installed on an iPhone device are provided by the underlying platform. The specific cryptographic implementation for the iOS platform can be found in the Apple iOS 16 Security Target documentation (VID11349) and CAVP Certificate #A3426

[iPadOS] TLS communication and S/MIME cryptographic services for Boxer application installed on an iPad device are provided by the underlying platform. The specific cryptographic implementation for the iPadOS platform can be found in the Apple iPadOS 16 Security Target documentation (VID11350) and CAVP Certificate #A3426

[Android] TLS communication cryptographic services for the Boxer application installed on a Samsung Galaxy device is provided by the underlying platform. The specific cryptographic implementation for the Android platform can be found in the Samsung Android 13 Security Target documentation (VID11342) and CAVP Certificate #A3285.

Additionally, when Boxer is installed on a device running the Android OS 10, the application includes OpenSSL software library 1.0.2zi to perform the cryptographic services for S/MIME functionality. The CAVP certificates for Boxer's OpenSSL Android implementation are specified in Table 6 in the ST.

SFR(s) Supported	Algorithm(s) (cryptographic operation)	Standard	CAVP Consolidated Cert. #
FCS_COP.1/KeyedHash FCS_CKM_EXT.5.3,	HMAC-SHA-256, 256-bit key size	NIST FIPS 198-1 and NIST FIPS 180-4	A5072
FCS_COP.1/SKC FCS_SMIME_EXT.1.2,	AES-128-CBC and AES-256-CBC	NIST SP 800-38A	A5072
FCS_COP.1/SKC Encryption of Boxer specific database used in support of FCS_STO_EXT.1(1) & (2) storage of specific keys.	AES-256-CBC	NIST SP 800-38A	A5072
FCS_COP.1/Hash FCS_SMIME_EXT.1.3,	SHA-256, SHA-384, SHA-512	NIST FIPS 180-4	A5072
FCS_COP.1/Sig FCS_SMIME_EXT.1.4,	RSA (2048, SHA-256)	NIST FIPS 186-4	A5072
FCS_RBG_EXT.2.1 (Android) per FCS_RBG_EXT.1.1 (Android)	DRBG CTR (AES-256)	NIST SP 800-90A	A5072
FCS_COP.1/SKC FCS_RBG_EXT.2.1 (Android) per FCS_RBG_EXT.1.1 (Android)	AES-256-CTR	NIST SP 800-38A	A5072

4.3.1.1 FCS_CKM_EXT.4

001	[EC_EP]FCS_CKM_EXT.4.1 – Cryptographic Key Destruction
Test Purpose:	The following test is only for key destruction provided by the email client: Test 1: For each type of authorization service, encryption mode and encryption operation, a known authorization factor, and chain of keys must be provided to the evaluator with an associated ciphertext data set (e.g. if a passphrase is used to create an intermediate key, then the ciphertext containing the encrypted key as well as the intermediate key itself must be provided to the evaluator.) The evaluator will use the email client in conjunction with a debugging or forensics utility to attempt to authorize themselves, resulting in the generation of a key or decryption of a key. The evaluator will ascertain from the TSS what the vendor defines as "no longer needed" and execute the sequence of actions via the email client to invoke this state. At this point, the evaluator should take a dump of volatile memory and search the retrieved dump for the provided authorization credentials or keys (e.g. if the password was "PaSSw0rd", perform a string search of the forensics dump for "PaSSw0rd"). The evaluator must document each command, program or action taken during this process, and must confirm that no plaintext keying material resides in volatile memory. The evaluator must perform this test three times to ensure repeatability. If during the course of this testing the evaluator finds that keying material remains in volatile memory, they should be able to identify the cause (i.e. execution of the grep command for "PaSSw0rd" caused a false positive) and document the reason for failure to comply with this requirement. The evaluator will repeat this same test, but looking for keying material in nonvolatile memory.
Test Instructions:	Manually execute this test per the test steps.
Test Procedures:	Android: Refer to “Android_Memory_Dump_Instructions.docx” in the “Test Evidence\FCS_CKM_EXT.4/Android” directory. iOS: Refer to “iOS_Memory_Dump_Instructions.docx” in the “Test Evidence\FCS_CKM_EXT.4/iOS” directory. iPadOS: Refer to “iPadOS_Memory_Dump_Instructions.docx” in the “Test Evidence\FCS_CKM_EXT.4/iPadOS” directory.
Test Results:	This activity passes as the evaluator observed for all keys that are destroyed by

	Boxer (the target application), there were no instances found in the collected volatile memory space comprising the Boxer application after the time of expected key destruction for those keys. Appropriate analysis was conducted to verify that for any detected keys, including intermediate keys as part of any wrapping operations were compared against the proprietary key wrapping specification.

4.3.1.2 FCS_CKM_EXT.5

002	**[EC_EP]FCS_CKM_EXT.5.1 – Cryptographic Key Derivation (Password/Passphrase Conditioning) – TD0266**
Test Purpose:	The evaluator shall compose passwords that either meet the requirements, or fail to meet the requirements, and shall verify that the TOE's behavior is consistent with the requirements. While the evaluator is not required (nor is it feasible) to test all possible compositions of passwords, the evaluator shall ensure that all characters, and minimum and maximum lengths listed in the requirement, are supported, and justify the subset of those characters chosen for testing. Support for Password/Passphrase characteristics: In addition to the analysis above, the evaluator shall also perform the following tests on a TOE configured according to the Operational Guidance. Test 1: Ensure that the TOE supports passwords/passphrases of 64 characters.
Test Instructions:	Manually execute this test per the test steps.
Test Procedures:	1. Attempt to assign a 64-character length password on the TOE: QAZWSXEDCRFVTGBYHNUJMIKOLP1029384756qazwsxedcrfvtgbyhnujmikolp12
Test Results:	This activity passes as the evaluator observed that a 64 character passcode was accepted when the second entry of the passcode was a match to the first.

003	**[EC_EP]FCS_CKM_EXT.5.1 – Cryptographic Key Derivation (Password/Passphrase Conditioning) – TD0266**
Test Purpose:	The evaluator shall compose passwords that either meet the requirements, or fail to meet the requirements, and shall verify that the TOE's behavior is consistent with the requirements. While the evaluator is not required (nor is it feasible) to test all possible compositions of passwords, the evaluator shall ensure that all characters, and minimum and maximum lengths listed in the requirement, are supported, and justify the subset of those characters chosen for testing. Support for Password/Passphrase characteristics: In addition to the analysis above, the evaluator shall also perform the following tests on a TOE configured according to the Operational Guidance. Test 2: Ensure that the TOE does not accept more than the maximum number of characters specified in FCS_CKM_EXT.5.1.
Test Instructions:	Manually execute this test per the test steps.
Test Procedures:	1. Attempt to assign a 513-character password on the TOE: a@uLPIPDNPk81ExNyzngCTLP2ofj8qovpgQjrNqVy3xndLbq7kiD0Wu6YsjfshiZdpAIFPbZnW5UAPyePhpGyB7DXbtvUxLpRl1NFN5SPjJoZpZJFAZCPMe7LyBzOlfYEIyiE8J0kouW2HSqDtVoLszHMOQ32CSwwL0IPJKz5U1gh0NV5aVa2tJo

	PTKXPQePa8Fm3PjlpJ9e8j9dQRt3lMP3Nx4BLXzC1yXeoHlIzdTekqupoYQlHuu MqPllrYvnLK5pSir6Gjr9fGeI9bZfTBOCXd7MOBxuZJjTaQnl9iyTXNxlufFXeO OE1CE5vlvUWc3Ltij8gEmnW3to9oAl3YbSvhECXvClknAgOygVPq2inp2IY9m0 QsMZBXS8M7Y0XTo2Ja8ue199RqRdN5N96hCqaRXFZC7SnyknxpvOPRDplSf UVZjEoD0AtzJiGcOruJSlvE2c4YsPrVidTYHpoXbz0Eaz8003NdPrVvTVzyxa4fg a5oQJ7nAfd8E9rC012
Test Results:	This activity passes as the evaluator observed that a 64 character passcode was accepted when the second entry of the passcode was a match to the first.

004	**[EC_EP]FCS_CKM_EXT.5.1 – Cryptographic Key Derivation (Password/Passphrase Conditioning) – TD0266**
Test Purpose:	The evaluator shall compose passwords that either meet the requirements, or fail to meet the requirements, and shall verify that the TOE's behavior is consistent with the requirements. While the evaluator is not required (nor is it feasible) to test all possible compositions of passwords, the evaluator shall ensure that all characters, and minimum and maximum lengths listed in the requirement, are supported, and justify the subset of those characters chosen for testing. Support for Password/Passphrase characteristics: In addition to the analysis above, the evaluator shall also perform the following tests on a TOE configured according to the Operational Guidance. Test 3: Ensure that the TOE does not accept less than the minimum number of characters specified in FCS_CKM_EXT.5.4. If the minimum length is settable by the administrator, the evaluator determines the minimum length or lengths to test.
Test Instructions:	Manually execute this test per the test steps.
Test Procedures:	1. Attempt to assign a 3-character password on the TOE: 123
Test Results:	This activity passes as the evaluator observed that a passcode of 3 characters was not accepted and the correct error or "your passcode should be a minimum 4 characters" was displayed. This is consistent with the administratively configurable setting set by the evaluator.

005	**[EC_EP]FCS_CKM_EXT.5.1 – Cryptographic Key Derivation (Password/Passphrase Conditioning) – TD0266**
Test Purpose:	The evaluator shall compose passwords that either meet the requirements, or fail to meet the requirements, and shall verify that the TOE's behavior is consistent with the requirements. While the evaluator is not required (nor is it feasible) to test all possible compositions of passwords, the evaluator shall ensure that all characters, and minimum and maximum lengths listed in the requirement, are supported, and justify the subset of those characters chosen for testing. Support for Password/Passphrase characteristics: In addition to the analysis above, the evaluator shall also perform the following tests on a TOE configured according to the Operational Guidance. Test 4: Ensure that the TOE supports passwords consisting of all characters listed in FCS_CKM_EXT.5.2 and of varying lengths within the range specified in FCS_CKM_EXT.5.4.
Test Instructions:	Manually execute this test per the test steps.
Test Procedures:	1. Attempt to assign the following password on the TOE (56-character

	length): 012345!@#$%^&*()GHIJKLMNOPQRSTUVWXYZghijklmnopqrstuvwxyz 2. Attempt to assign the following password on the TOE (26-character length): ^&*456789ExtraLongP@ssw0rd 3. Attempt to assign the following password on the TOE (16-character length): 1234!#$%()ABCDEFabcdef 4. Attempt to assign the following password on the TOE (10-character length): P@ssw0rdP@
Test Results:	This activity passes as the evaluator observed that a 56-character, 26 character, 16 character and a 10 character password successfully work using upper, lower, the defined special characters, and numbers.

4.3.1.3 FCS_RBG_EXT.1(Android), (iOS), & (iPadOS)

006	**[APP_PP]FCS_RBG_EXT.1.1(Android), (iOS), & (iPadOS) – Random Bit Generation Services**
Test Purpose:	If "invoke platform-provided DRBG functionality" is selected, the following tests shall be performed: The evaluator shall decompile the application binary using a decompiler suitable for the application (TOE). The evaluator shall search the output of the decompiler to determine that, for each API listed in the TSS, that API appears in the output. If the representation of the API does not correspond directly to the strings in the following list, the evaluator shall provide a mapping from the decompiled text to its corresponding API, with a description of why the API text does not directly correspond to the decompiled text and justification that the decompiled text corresponds to the associated API. The following are the per-platform list of acceptable APIs: Platforms: Android... The evaluator shall verify that the application uses at least one of javax.crypto.KeyGenerator class or the java.security.SecureRandom class or /dev/random or /dev/urandom. Platforms: Apple iOS... The evaluator shall verify that the application invokes either SecRandomCopyBytes, CCRandomGenerateBytes, or CCRandomCopyBytes, or uses /dev/random directly to acquire random.
Test Instructions:	Manually execute this test per the test steps.
Test Procedures:	Android: 1. Search for all instances of /dev/random and /dev/urandom in the TOE

	source code. iOS / iPadOS: 1. Perform a decompile/static source code analysis. 2. Verify that the TOE invokes SecRandomCopyBytes or uses /dev/random directory to acquire random.
Test Results:	This activity passes as the evaluator confirmed, during a source code review of the Android Boxer code, that the TOE invokes /dev/random and /dev/urandom to acquire random. Additionally, the evaluator confirmed, during a source code review of the iOS/iPadOS code that the TOE invokes SecRandCopyBytes to acquire random.

4.3.1.4 FCS_SMIME_EXT.1

007	**[EC_EP]FCS_SMIME_EXT.1 – Secure/Multipurpose Internet Mail Extensions (S/MIME)**
Test Purpose:	These tests can be performed in conjunction with the tests defined in FIA_X509 for certificate/certificate chain verification and FDP_NOT_EXT.1. Test 1: The evaluator shall both send and receive a message with no protection (no signature or encryption) and verify that the message is transmitted properly and can be viewed at the receiving agent. This transmission can be performed as part of a number of mechanisms; it is sufficient to observe that the message arrives at the intended recipient with the same content as when sent.
Test Instructions:	Manually execute this test per the test steps.
Test Procedures:	Sending E-mail 1. Using the TOE, create and send an e-mail with no protection (no signature nor encryption) to the remote test recipient. 2. Verify the e-mail sent in Step 1 is transmitted and properly received by the remote test recipient. Receiving E-mail 1. Using the remote test e-mail client, create and send an e-mail with no protection (no signature nor encryption) to the TOE. 2. Verify the e-mail sent in Step 1 is transmitted and properly received by the TOE. 3. Verify that no notifications are present upon viewing the received e-mail.
Test Results:	This activity passes as the evaluator observed that all platforms (Android, iOS, iPad) could send and receive an email with no protection (signature or encryption). Both the sent and received emails were verified to ensure that the content was consistent and did not have protections icons displayed.

008	**[EC_EP]FCS_SMIME_EXT.1 – Secure/Multipurpose Internet Mail Extensions (S/MIME)**
Test Purpose:	These tests can be performed in conjunction with the tests defined in FIA_X509 for certificate/certificate chain verification and FDP_NOT_EXT.1. Test 2: The evaluator shall both send and receive a signed message using each of the algorithms specified in the ST corresponding to the requirement and verify that the signature is valid for both received and sent messages.

	After verifying the signatures are valid, the evaluator shall send a signed message using each of the algorithms specified in the ST and use a man-in-the-middle tool to modify at least one byte of the message such that the signature is no longer valid. This can be done by modifying the content of the message over which the signature is calculated or by modifying the signature itself. The evaluator shall verify that the received message fails the signature validation check.
Test Instructions:	Manually execute this test per the test steps.
Test Procedures:	Sending E-mail 1. Using the TOE, create and send a signed e-mail with no encryption using the sha256withRSAEncryption (sha256 message digest) signature algorithm to the remote test recipient. 2. Verify the e-mail sent in Step 1 is transmitted, properly received, and that the signature is successfully verified by the remote test recipient. 3. Repeat Steps 1-2, except in Step 1, use the sha256withRSAEncryption (sha384 message digest) signature algorithm. 4. Repeat Steps 1-3, except in Step 1, use the sha256withRSAEncryption (sha512 message digest) signature algorithm. Receiving E-mail 1. Using the remote test e-mail client, create and send a signed e-mail with no encryption using the sha256withRSAEncryption (sha256 message digest) signature algorithm to the TOE. 2. Verify the e-mail sent in Step 1 is transmitted, properly received by the TOE, and that the signature is successfully verified by the TOE. 3. Verify that the TOE displays the signed message notification upon viewing. 4. Repeat Steps 1-3, except in Step 1, use the sha256withRSAEncryption (sha384 message digest) signature algorithm. 5. Repeat Steps 1-3, except in Step 1, use the sha256withRSAEncryption (sha512 message digest) signature algorithm. Receiving E-mail (man-in-the-middle) 1. Using the test machine e-mail client, create and send a signed e-mail with no encryption using the sha256withRSAEncryption (sha256 message digest) signature algorithm to the TOE. 2. Using the MITM tool, modify at least one byte of the e-mail message over which the signature was calculated. 3. Verify the e-mail sent in Step 1 is transmitted, able to be read, and that the signature fails to be verified by the TOE. 4. Repeat Steps 1-3, except in Step 1, use the sha256withRSAEncryption (sha384 message digest) signature algorithm. 5. Repeat Steps 1-3, except in Step 1, use the sha256withRSAEncryption (sha512 message digest) signature algorithm.
Test Results:	This activity passes as the evaluator observed that all platforms (Android, iOS, iPad) could send and receive an email with 256, 384, and 512 signature protection and no encryption. Both the sent and received emails were verified to ensure that

	the content was consistent and did show the signature icon. The evaluator also observed that emails with 256, 384, and 512 signature protection and no encryption that have been tampered with fails the signature validation check and displays a "This message has been tampered with" to the TOE user recipient and shows the encryption icon in red.

009	**[EC_EP]FCS_SMIME_EXT.1 – Secure/Multipurpose Internet Mail Extensions (S/MIME) – TD0560**
Test Purpose:	These tests can be performed in conjunction with the tests defined in FIA_X509 for certificate/certificate chain verification and FDP_NOT_EXT.1. Test 3: a) The evaluator shall send an encrypted message from the TOE to an OE receiver using each of the algorithms specified in the ST. The evaluator shall verify that each message is encrypted and the OE receiver can successfully decrypt each message. b) The evaluator shall use the OE receiver to send an encrypted reply back to the TOE for each message sent in a). The evaluator shall verify that each reply is encrypted and the TOE can successfully decrypt each reply.
Test Instructions:	Manually execute this test per the test steps.
Test Procedures:	Sending E-mail 1. Using the TOE, create and send an unsigned, protected e-mail using the AES-128 CBC encryption algorithm to the remote test recipient. 2. Verify the e-mail sent in Step 2 is encrypted in transit, properly received, and that it is successfully decrypted by the remote test recipient. 3. Repeat Steps 1-4, except replace “AES-128 CBC” with “AES-256 CBC” in Step 2. Receiving E-mail 1. Using the remote test e-mail client, create and send an unsigned, protected response e-mail using the AES-128 CBC encryption algorithm to the TOE. 2. Verify the e-mail sent in Step 2 is encrypted in transit, properly received by the TOE, and that it is successfully decrypted by the TOE. 3. Verify that the TOE displays the encrypted message notification upon viewing. 4. Repeat Steps 1-4 , except replace “AES-128 CBC” with “AES-256 CBC” in Step 2.
Test Results:	This activity passes as the evaluator observed that all platforms (Android, iOS, iPad) could send and receive an email with aes-128-cbc and aes-256-cbc encryption and no signature. Both the sent and received emails were verified to ensure that the content was consistent and did show the encrypted icon.

010	**[EC_EP]FCS_SMIME_EXT.1 – Secure/Multipurpose Internet Mail Extensions (S/MIME)**
Test Purpose:	These tests can be performed in conjunction with the tests defined in FIA_X509 for certificate/certificate chain verification and FDP_NOT_EXT.1. Test 4: The evaluator shall both send and receive a message that is both signed and

	encrypted. In addition, the evaluator shall use a man-in-the-middle tool to modify at least one byte of the message such that the encryption and signature are no longer valid. The evaluator shall verify that the received message fails to decrypt, fails the signature validation check, and/or both.
Test Instructions:	Manually execute this test per the test steps.
Test Procedures:	Sending E-mail 1. Using the TOE, create and send a signed, protected e-mail using the sha256withRSAEncryption signature algorithm (sha256 message digest) and the AES-128 CBC encryption algorithm to the remote test recipient. 2. Verify the e-mail sent in Step 2 is encrypted in transit, properly received, that it is successfully decrypted, and that the signature is validated by the remote test recipient. Receiving E-mail 1. Using the remote test e-mail client, create and send signed, protected e-mail using the sha256withRSAEncryption signature algorithm (sha256 message digest) and the AES-128 CBC encryption algorithm to the TOE. 2. Verify the e-mail sent in Step 2 is encrypted in transit, properly received by the TOE, that it is successfully decrypted and that the signature is validated by the TOE. 3. Verify that the TOE displays the encrypted and signed message notification upon viewing. Receiving E-mail (man-in-the-middle) 1. Using the test machine e-mail client, create and send a signed, protected e-mail using the AES-128 CBC encryption algorithm to the TOE. 2. Using the MITM tool, modify at least one byte of the e-mail message such that the encryption and signature are no longer valid. 3. Verify the e-mail sent in Step 1 is encrypted in transit, unable to be read nor decrypted.
Test Results:	This activity passes as the evaluator observed that all platforms (Android, iOS, iPad) could send and receive an email with 256 signature protection and aes-128-cbc encryption. Both the sent and received emails were verified to ensure that the content was consistent and did show the encryption protection and signature icons. The evaluator also observed that emails with 256 signature protection and aes-128-cbc encryption that have been tampered with fails the signature validation check and displays a "Could not process message: unknown error" (Android) or "Unable to locate certificate to decrypt this message or the message is corrupted" (iOS/iPad) to the TOE user recipient and does not display contents of email.

011	**[EC_EP]FCS_SMIME_EXT.1 – Secure/Multipurpose Internet Mail Extensions (S/MIME)**
Test Purpose:	These tests can be performed in conjunction with the tests defined in FIA_X509 for certificate/certificate chain verification and FDP_NOT_EXT.1. Test 5: The evaluator shall send a signed message to the email client using a

	signature algorithm not supported according to the digestAlgorithm ID (e.g., SHA1). The evaluator shall verify that the email client provides a notification that the contents cannot be verified because the signature algorithm is not supported.
Test Instructions:	Manually execute this test per the test steps.
Test Procedures:	Receiving E-mail 1. Using the remote test e-mail client, create and send signed, unprotected e-mail using the SHA1 digestAlgorithm ID to the TOE. 2. Verify the e-mail sent in Step 2 is transmitted, received by the TOE, and that the signature cannot be verified. 3. Verify that the TOE displays a notification that the contents cannot be verified because the signature algorithm is not supported.
Test Results:	This activity passes as the evaluator observed that the unsupported sha1 signature algorithm resulted in a failure to validate the email, an error being displayed "Signing algorithm is not compliant" to the TOE user recipient, and the displaying the contents of the email.

012	**[EC_EP]FCS_SMIME_EXT.1 – Secure/Multipurpose Internet Mail Extensions (S/MIME) – TD0560**
Test Purpose:	These tests can be performed in conjunction with the tests defined in FIA_X509 for certificate/certificate chain verification and FDP_NOT_EXT.1. Test 6: The evaluator shall send an encrypted message to the email client using an encryption algorithm not supported according to the signatureAlgorithm field. The evaluator shall verify that the email client does not display/decrypt the contents of the message.
Test Instructions:	Manually execute this test per the test steps.
Test Procedures:	Receiving E-mail 1. Using the remote test e-mail client, create and send an unsigned, encrypted e-mail using the AES-192 CBC encryption algorithm to the TOE. 2. Verify the e-mail sent in Step 1 is transmitted, received by the TOE, and that the TOE does not display nor decrypt the contents of the message.
Test Results:	This activity passes as the evaluator observed that the unsupported aes_192_cbc encryption algorithm resulted in a failure to decrypt the email, an error being displayed "Encryption algorithm is not compliant" to the TOE user recipient, and the contents of the email are not displayed.

013	**[EC_EP]FCS_SMIME_EXT.1 – Secure/Multipurpose Internet Mail Extensions (S/MIME)**
Test Purpose:	These tests can be performed in conjunction with the tests defined in FIA_X509 for certificate/certificate chain verification and FDP_NOT_EXT.1. Test 7: The evaluator shall send the email client a message signed by a certificate without the digitalSignature bit set. The evaluator shall verify that the email client notifies the user that the signature is invalid.
Test Instructions:	Manually execute this test per the test steps.
Test Procedures:	Receiving E-mail 1. Using the remote test e-mail client, create and send signed, unprotected e-

	mail using the sha256withRSAEncryption signature algorithm (sha256 message digest) signed with a certificate without the digitalSignature bit set to the TOE. 2. Verify the e-mail sent in Step 1 is transmitted, received by the TOE, and that the signature is invalid.
Test Results:	This activity passes as the evaluator observed that an email without the digital signature bit set correctly results in an error being displayed "Sender's certificate is not compliant" (Android) or "Sender's certificate is not valid" (iOS/iPad) and the contents of the email displayed.

014	**[EC_EP]FCS_SMIME_EXT.1 – Secure/Multipurpose Internet Mail Extensions (S/MIME)**
Test Purpose:	These tests can be performed in conjunction with the tests defined in FIA_X509 for certificate/certificate chain verification and FDP_NOT_EXT.1. Test 8: The evaluator shall send the email client a message signed by a certificate without the Email Protection purpose in the extendedKeyUsage. The evaluator shall verify that the email client notifies the user that the signature is invalid.
Test Instructions:	Manually execute this test per the test steps.
Test Procedures:	Receiving E-mail 1. Using the remote test e-mail client, create and send signed, unprotected e-mail using the sha256withRSAEncryption signature algorithm (sha256 message digest) signed with a certificate without the Email Protection purpose present in the extendedKeyUsage field to the TOE. 2. Verify the e-mail sent in Step 1 is transmitted, received by the TOE, and that the signature is invalid.
Test Results:	This activity passes as the evaluator observed that an email without the Email Protection purpose in the extendedKeyUsage results in an error being displayed "Sender's certificate is not compliant" (Android) or "Sender's certificate is not valid" (iOS/iPad) and the contents of the email displayed.

015	**[EC_EP]FCS_SMIME_EXT.1 – Secure/Multipurpose Internet Mail Extensions (S/MIME)**
Test Purpose:	These tests can be performed in conjunction with the tests defined in FIA_X509 for certificate/certificate chain verification and FDP_NOT_EXT.1. Test 9: The evaluator shall verify that the email client uses OCSP or downloads the CRL at the assigned frequency.
Test Instructions:	Manually execute this test per the test steps.

Test Procedures:	1. Configure (refer to AGD supplement Section 6.3.1) the TOE application to perform OCSP checking with the following Key Value Pairs configured in UEM for the TOE:
	
	Android:
	
	`PolicySMIMEEnableRevocationCheck == 1`
	`PolicySMIMERevocationTTL == 0`
	
	iOS / iPadOS:
	
	`PolicySMIMEEnableRevocationCheck == 1`

	`PolicySMIMERevocationTTL == 0`
	`PolicySMIMERevocationUseAIA == 2`
	
	2. Send a signed e-mail to the TOE.
	3. Begin capturing packets between the TOE and the remote OCSP server.
	4. On the TOE, open the e-mail sent in Step 2.
	5. Stop capturing packets between the TOE and the remote OCSP server.
	6. Verify that the TOE verifies the certificate used to validate the signature of the received e-mail via OCSP at the assigned frequency:
	a. Close and re-open the signed e-mail immediately after the expected amount of time.

Test Results:	This activity passes as the evaluator observed that the CRL was obtained at the assigned frequency.

4.3.1.5 FCS_STO_EXT.1(1) & (2)

016	**[APP_PP]FCS_STO_EXT.1.1(1) & (2) – Storage of Secrets**
Test Purpose:	For all credentials for which the application implements functionality, the evaluator shall verify credentials are encrypted according to FCS_COP.1/SKC or conditioned according to FCS_CKM.1.1/AK and FCS_CKM.1/PBKDF. For all credentials for which the application invokes platform-provided functionality, the evaluator shall perform the following actions which vary per platform. Platforms: Android... The evaluator shall verify that the application uses the Android KeyStore or the Android KeyChain to store certificates. Platforms: Apple iOS... The evaluator shall verify that all credentials are stored within a Keychain.
Test Instructions:	Manually execute this test per the test steps.
Test Procedures:	Android: 1. For all credentials where the TOE invokes platform-provided functionality, perform a static code analysis. 2. Verify that the TOE uses the Android KeyStore or the Android KeyChain to store certificates. 3. For all credentials where the TOE implements functionality for FCS_STO_EXT.1(1) and (2), verify that it is encrypted according to FCS_COP.1/SKC. iOS / iPadOS: 1. For all credentials where the TOE invokes platform-provided functionality, perform a static code analysis. 2. Verify that the TOE uses an iOS Keychain. (For completeness, the secure enclave is considered a keychain in order to validate all defined keys.) 3. For all credentials where the TOE implements functionality for FCS_STO_EXT.1(1) & (2), verify that it is encrypted according to FCS_COP.1/SKC.
Test Results:	This activity passes as the evaluator observed, during a code review of the:

	Boxer Android code, the storage of specific persistent keys (that are applicable from FCS_STO_EXT.1(1)) are stored either in the Boxer database or Keychain as specified in the ST. iOS / iPadOS code, the storage of specific persistent keys (that are applicable from FCS_STO_EXT.1(1)) are stored either in the Boxer database or in the iOS Keychain as specified in ST. Tables from the ST and results of the code review were found to be consistent.

4.3.2 User Data Protection

4.3.2.1 FDP_DAR_EXT.1

017	**[APP_PP]FDP_DAR_EXT.1.1 - Encryption Of Sensitive Application Data – TD0756**
Test Purpose:	Evaluation activities (after the identification of the sensitive data) are to be performed on all sensitive data listed that are not covered by FCS_STO_EXT.1. If "implement functionality to encrypt sensitive data as defined in the PP-Module for File Encryption" or "protect sensitive data in accordance with FCS_STO_EXT.1" is selected, the evaluator shall inventory the filesystem locations where the application may write data. The evaluator shall run the application and attempt to store sensitive data. The evaluator shall then inspect those areas of the filesystem to note where data was stored (if any), and determine whether it has been encrypted. If "leverage platform-provided functionality" is selected, the evaluation activities will be performed as stated in the following requirements, which vary on a per-platform basis. Platforms: Android... The evaluator shall inspect the TSS and verify that it describes how files containing sensitive data are stored with the MODE_PRIVATE flag set. Platforms: Apple iOS... The evaluator shall inspect the TSS and ensure that it describes how the application uses the Complete Protection, Protected Unless Open, or Protected Until First User Authentication Data Protection Class for each data file stored locally.
Test Instructions:	Manually execute this test per the test steps.
Test Procedures:	Android: Examine the TOE application source code for all instances where files containing sensitive data are stored. Verify that the method(s) called pass the “MODE_PRIVATE” flag as the parameter. Exercise application and attempt to store sensitive information. Attempt to locate within filesystem and determine if the information has been stored unencrypted. Verify findings are consistent with TSS. iOS / iPadOS: Examine the TOE application source code for all instances where files containing

	sensitive data are stored. Verify that string searches for “NSDataWritingFileProtectionComplete”, “NSDataWritingFileProtectionCompleteUnlessOpen”, or “NSDataWritingFileProtectionCompleteUntilFirstUserAuthentication” are present in the TOE source code AND “NSDataWritingFileProtectionMask.” and “NSDataWritingFileProtectionNone”, are not present in the TOE source code Exercise application and attempt to store sensitive information. Attempt to locate within filesystem and determine if the information has been stored unencrypted. Verify findings are consistent with TSS.
Test Results:	This activity passes as the evaluator observed that for: Android: The code review found that all files containing sensitive data (calendar, address book (i.e., contacts), system accounts, and profiles) are stored using the getSharedPreferences(String name, int mode) method with the "MODE_PRIVATE" flag set. Additionally, for Android, all instances of the user attempting to save contact or calendar information is stored on the device encrypted as a result of the file-based encryption provided by the OS. This is consistent with the following description from the TSS: "The Android platform automatically enforces file-based encryption (FBE) using AES-256-XTS encryption. The TOE’s file creation scheme requires sensitive data files to be saved with the MODE_PRIVATE flag set. All instances where files containing sensitive data are stored calling the getSharedPreferences(String name, int mode) method (which is an overridden method defined in the Boxer application source code) with the “MODE_PRIVATE” flag as the second parameter. Sensitive data includes: calendar, address book (i.e., contacts), system accounts, and profiles which are stored in the internal Boxer database that is fully encrypted in addition to the file encryption (double encryption)." The code review findings were consistent with TSS descriptions for Android using “MODE_PRIVATE”. iOS / iPad: The code review found that all files containing sensitive data (calendar and address book (i.e., contacts), are stored using NSFileProtectionCompleteUntilFirstUserAuthentication. Additionally, for iOS/iPad, all instances of the user attempting to save contact or calendar information is stored on the device encrypted as a result of the file-based encryption. This is consistent with the following description from the TSS: "The TOE’s file creation scheme requires sensitive data files to be saved with the MODE_PRIVATE flag set. All instances where files containing sensitive data are stored calling the getSharedPreferences(String name, int mode) method (which is an overridden method defined in the Boxer application source code) with the “MODE_PRIVATE” flag as the second parameter. Sensitive data includes: calendar, address book (i.e., contacts), system accounts, and profiles which are stored in the internal Boxer database that is fully encrypted in addition to the file encryption (double encryption)." Assessing each of the applicable NSDataWritingFileProtection* options, the evaluator was able to determine only

	the NSFileProtectionCompleteUntilFirstUserAuthentication was used. The code review findings were consistent with TSS descriptions for iOS using "NSFileProtectionCompleteUntilFirstUserAuthentication".

4.3.2.2 FDP_DEC_EXT.1(Android), (iOS), & (iPadOS)

018	**[APP_PP]FDP_DEC_EXT.1.1(Android), (iOS), & (iPadOS) – Access to Platform Resources**
Test Purpose:	Platforms: Android... The evaluator shall verify that each uses-permission entry in the AndroidManifest.xml file for access to a hardware resource is reflected in the selection. Platforms: Apple iOS... The evaluator shall verify that either the application or the documentation provides a list of the hardware resources it accesses.
Test Instructions:	Manually execute this test per the test steps.
Test Procedures:	Android: 1. Execute the following command to parse the TOE's AndroidManifest.xml file: aapt2 dump permissions Boxer.apk 2. Verify that each "<uses-permission>" entry in the output for access to a hardware resource is reflected in the selection for FDP_DEC_EXT.1.1(Android). iOS / iPadOS: Verify that the TOE application and/or its documentation lists the hardware platform resource access permissions that are requested by the application.

Test Results:	This activity passes as the evaluator observed: Android: For each entry in "AndroidManifest.xml" that lists a "uses-permission" instance, a determination was made to assess whether it was first considered a hardware resource. If so, then it was compared with the set of selected and assigned hardware resources in the SFR from the ST. Otherwise, if not a hardware resource it was marked as such. Each "uses-permission" entry was either a hardware resource that was listed in the SFR in the ST or it was determined to not be a hardware resource. There were no unaccounted for hardware resources in the AndroidManifest.xml file. Hardware resources found in the AndroidManifest.xml: network connectivity, camera, NFC, device storage, phone, fingerprint, and vibrator. These matched the list in the ST and AGD Section 7.4. iOS / iPadOS: Section 7.4 of the AGD lists the hardware resources that are requested by the TOE. The list of hardware resources in the AGD is consistent with the selections from the ST. For each hardware resource, the AGD provides a justification for why access is required. For example, device storage is requested to have the ability to save email attachments and access files to attach to the emails. The list of hardware resources was consistent with the hardware resources discovered in AndroidManifest.xml (Android) and guidance documentation (iOS/iPadOS).

019	**[APP_PP]FDP_DEC_EXT.1.2(Android), (iOS), & (iPadOS) – Access to Platform Resources**
Test Purpose:	Platforms: Android... The evaluator shall verify that each uses-permission entry in the AndroidManifest.xml file for access to a sensitive information repository is reflected in the selection. Platforms: Apple iOS... The evaluator shall verify that either the application software or its documentation provides a list of the sensitive information repositories it accesses.
Test Instructions:	Manually execute this test per the test steps.
Test Procedures:	Android: Execute the following command to parse the TOE's AndroidManifest.xml file: aapt2 dump permissions Boxer.apk Verify that each "<uses-permission>" entry in the output for access to a sensitive information repository is reflected in the selection for FDP_DEC_EXT.1.2(Android). iOS / iPadOS: Verify that the TOE application and/or its documentation lists the sensitive platform resource access permissions that are requested by the application.
Test Results:	This activity passes as the evaluator observed: Android: For each entry in " AndroidManifest.xml" that lists a "uses-permission" instance, a

	determination was made to assess whether it was first considered a sensitive information repository. If so, then it was compared with the set of selected and assigned sensitive information repositories in the SFR from the ST. Otherwise, if not a sensitive information repository, it was marked as such. Each "uses-permission" entry was either a sensitive information repository that was listed in the SFR in the ST or it was determined to not be a sensitive information repository. There were no unaccounted for sensitive information repositories in the AndroidManifest.xml file. Sensitive information repositories found in the AndroidManifest.xml: Address book, Calendar, Accounts and Profile. These matched the list in the ST and AGD Section 7.4. iOS / iPadOS: Section 7.4 of the AGD lists the sensitive information repositories that are requested by the TOE. The list of sensitive information repositories in the AGD is consistent with the selections from the ST. For each sensitive information repository, the AGD provides a justification for why access is required. For example, calendar access is requested to add or modify calendar events and send email to guests. The list of sensitive information repositories was consistent with sensitive information repositories discovered in AndroidManifest.xml (Android) and guidance documentation (iOS)

4.3.2.3 FDP_NET_EXT.1

020	**[APP_PP]FDP_NET_EXT.1.1 – Network Communications**
Test Purpose:	The evaluator shall perform the following tests: Test 1: The evaluator shall run the application. While the application is running, the evaluator shall sniff network traffic ignoring all non-application associated traffic and verify that any network communications witnessed are documented in the TSS or are user-initiated.
Test Instructions:	Manually execute this test per the test steps.
Test Procedures:	Android: 1. Begin capturing packets from the Wireless Access Point mirror port that the TOE device is connected. 2. Launch the TOE application. 3. Perform some activity on the TOE, such as sending an e-mail, force e-mail sync with server, e-mail search, and performing a Global Address List (GAL) lookup. 4. Execute the following command to list network connections on the TOE device: netstat -tulpna 5. Stop capturing packets from the Wireless Access Point mirror port that the TOE device is connected. 6. Inspect the packet capture and verify that any application associated network communications witnessed are documented in the TSS or are user-initiated.

	iOS / iPadOS: 1. Begin capturing packets from and to the TOE using Wireshark on the test machine. 2. Launch the TOE application via Instruments from the test machine. 3. Perform some activity on the TOE, such as sending an e-mail, force e-mail sync with server, e-mail search, and performing a Global Address List (GAL) lookup. 4. Stop monitoring the TOE application with Instruments. 5. Stop capturing packets from and to the TOE. 6. Inspect the packet capture and verify that any application associated network communications witnessed are documented in the TSS or are user-initiated.
Test Results:	This activity passes as the evaluator observed: Android: The first launch packet capture and netstat output file correlate the TCP sockets that were established (initiated by the application/user) upon first launch after a fresh installation of Boxer. Per the security target TSS section, there were application associated sockets opened on local ephemeral TCP ports to remote host for the remote mail server (10.137.2.71, TCP/443), and other foreign IP addresses (Amazon) associated with the TOE application. iOS/iPadOS: The first launch packet capture and Xcode Instruments output files correlate the TCP sockets that were established (initiated by the application/user) upon first launch after a fresh installation of Boxer. Per the security target TSS section, there were application associated sockets opened on local ephemeral TCP ports to remote host for the remote mail server (10.137.2.71, TCP/443), and other foreign IP addresses (Amazon, Akamai) associated with the TOE application.

021	**[APP_PP]FDP_NET_EXT.1.1 – Network Communications**
Test Purpose:	The evaluator shall perform the following tests: Test 2: The evaluator shall run the application. After the application initializes, the evaluator shall run network port scans to verify that any ports opened by the application have been captured in the ST for the third selection and its assignment. This includes connection-based protocols (e.g. TCP, DCCP) as well as connectionless protocols (e.g. UDP).
Test Instructions:	Manually execute this test per the test steps.
Test Procedures:	Android: 1. Launch the TOE application. 2. After the application is initialized, execute the following command to initiate a network-based port scan against the TOE: netstat -tulpna 3. Verify that any ports opened by the TOE have been captured in the ST for the third selection and its assignment for this test assurance activity

	SFR. iOS / iPadOS: 1. Launch the TOE application via Instruments from the test machine. 2. After the application is initialized, stop monitoring the TOE application with Instruments. 3. Verify that any ports opened by the TOE have been captured in the ST for the third selection and its assignment for this test assurance activity SFR.
Test Results:	This activity passes as the evaluator observed: Android: The launch packet capture and netstat output file correlate the TCP sockets that were established (initiated by the application/user) as part of regular use of Boxer. Per the security target TSS section, there were application associated sockets opened on local persistent TCP ports to remote host for the remote mail server (10.137.2.71, TCP/443), and other foreign IP addresses (Amazon) associated with the TOE application. iOS/iPadOS: The launch packet capture and Xcode instruments output files correlate the TCP sockets that were established (initiated by the application/user) as part of regular use of Boxer. Per the security target TSS section, there were application associated sockets opened on local persistent TCP ports to remote host for the remote mail server (10.137.2.71, TCP/443), and other foreign IP addresses (Amazon, Akamai) associated with the TOE application.

022	**[APP_PP]FDP_NET_EXT.1.1 – Network Communications**
Test Purpose:	The evaluator shall perform the following tests: Platforms: Android... If "no network communication" is selected, the evaluator shall ensure that the application's AndroidManifest.xml file does not contain a uses-permission or uses-permission-sdk-23 tag containing android:name="android.permission.INTERNET". In this case, it is not necessary to perform the above Tests 1 and 2, as the platform will not allow the application to perform any network communication.
Test Instructions:	Manually execute this test per the test steps.
Test Procedures:	*N/A – "no network communication" is not selected.*
Test Results:	This activity is considered satisfied.

4.3.2.4 FDP_NOT_EXT.1

023	**[EC_EP]FDP_NOT_EXT.1.1 – Notification of S/MIME Status**
Test Purpose:	The evaluator shall perform the following tests and may perform them in conjunction with the tests for FCS_SMIME_EXT.1: Test 1: The evaluator shall send the client an unencrypted and unsigned email and verify that no notifications are present upon viewing.
Test Instructions:	Manually execute this test per the test steps.
Test Procedures:	NOTE: This test is performed in conjunction with [EC_PP]FCS_SMIME_EXT.1

	– Test Case 007.
Test Results:	This activity passes as the evaluator observed for all platforms that an email with no signature and no encryption showed no symbols.

024	**[EC_EP]FDP_NOT_EXT.1.1 – Notification of S/MIME Status**
Test Purpose:	The evaluator shall perform the following tests and may perform them in conjunction with the tests for FCS_SMIME_EXT.1: Test 2: The evaluator shall send the client an encrypted email and verify that the encrypted notification is present upon viewing.
Test Instructions:	Manually execute this test per the test steps.
Test Procedures:	*NOTE: This test is performed in conjunction with [EC_PP]FCS_SMIME_EXT.1 – Test Case 009.*
Test Results:	This activity passes as the evaluator observed, for all platforms, that an encrypted email showed the padlock icon.

025	**[EC_EP]FDP_NOT_EXT.1.1 – Notification of S/MIME Status**
Test Purpose:	The evaluator shall perform the following tests and may perform them in conjunction with the tests for FCS_SMIME_EXT.1: Test 3: The evaluator shall send the client a valid signed email and verify that the signed notification is present upon viewing.
Test Instructions:	Manually execute this test per the test steps.
Test Procedures:	*NOTE: This test is performed in conjunction with [EC_PP]FCS_SMIME_EXT.1 – Test Case 008.*
Test Results:	This activity passes as the evaluator observed, for all platforms, that a signed email showed the signature icon.

026	**[EC_EP]FDP_NOT_EXT.1.1 – Notification of S/MIME Status**
Test Purpose:	The evaluator shall perform the following tests and may perform them in conjunction with the tests for FCS_SMIME_EXT.1: Test 4: The evaluator shall send the client an invalid signed email (for example, using a certificate that does not contain the correct email address or a certificate that does not chain to the root store) and verify that the invalid signature notification is present upon viewing.
Test Instructions:	Manually execute this test per the test steps.
Test Procedures:	1. From the test e-mail client, send the TOE an e-mail that is signed using an e-mail address that does not correspond to the identity of the certificate used to sign it. 2. Verify that the e-mail is received by the TOE and that upon viewing it, the TOE displays the invalid signature notification.
Test Results:	This activity passes as the evaluator observed for all platforms that when the TOE receives an e-mail that is signed using an e-mail address that does not correspond to the identity of the certificate used to sign it, the TSF marks/displays the email with a “Sender's certificate mismatched” notice.

4.3.2.5 FDP_SMIME_EXT.1

027	**[EC_EP]FDP_SMIME_EXT.1.1 – S/MIME**

Test Purpose:	Tests for this element are performed in conjunction with tests for FCS_SMIME_EXT.1 and FDP_NOT_EXT.1
Test Instructions:	Manually execute this test per the test steps.
Test Procedures:	*N/A – Tests for this element are performed in conjunction with tests for FCS_SMIME_EXT.1 and FDP_NOT_EXT.1.*
Test Results:	This activity is considered satisfied.

4.3.3 Identification and Authentication

4.3.3.1 FIA_X509_EXT.1

028	**[APP_PP]FIA_X509_EXT.1.1 – X.509 Certificate Validation**
Test Purpose:	The tests described must be performed in conjunction with the other certificate services evaluation activities, including the functions in FIA_X509_EXT.2.1. The tests for the extendedKeyUsage rules are performed in conjunction with the uses that require those rules. If the application supports chains of length four or greater, the evaluator shall create a chain of at least four certificates: the node certificate to be tested, two Intermediate CAs, and the self-signed Root CA. If the application supports a maximum trust depth of two, then a chain with no Intermediate CA should instead be created. Test 1: The evaluator shall demonstrate that validating a certificate without a valid certification path results in the function failing, for each of the following reasons, in turn: • by establishing a certificate path in which one of the issuing certificates is not a CA certificate, • by omitting the basicConstraints field in one of the issuing certificates, • by setting the basicConstraints field in an issuing certificate to have CA=False, • by omitting the CA signing bit of the key usage field in an issuing certificate, and • by setting the path length field of a valid CA field to a value strictly less than the certificate path. The evaluator shall then establish a valid certificate path consisting of valid CA certificates, and demonstrate that the function succeeds. The evaluator shall then remove trust in one of the CA certificates, and show that the function fails.
Test Instructions:	Manually execute this test per the test steps.
Test Procedures:	(a) Issuing certificate not a CA certificate: 1. Configure the remote entity to present a certificate chain to the TOE such that the issuing certificate of the remote node entity certificate is not a CA certificate. 2. Begin capturing packets between the TOE and the remote entity. 3. Cause the TOE to establish a connection to the remote entity. 4. Stop capturing packets between the TOE and the remote entity. 5. Verify the connection is unsuccessful. (b) Omitting the basicConstraints field in one of the issuing certificates: NOTE: This is tested in [APP_PP]FIA_X509_EXT.1 – Test Case 037.

	(c) Setting the basicConstraints field in an issuing certificate to have CA=False: NOTE: This is tested in [APP_PP]FIA_X509_EXT.1 – Test Case 038. (d) Omitting the CA signing bit of the key usage field in an issuing certificate: 1. Configure the remote entity to present a certificate chain to the TOE such that the issuing certificate of the remote node entity certificate does not have the CA signing bit in the key usage field. 2. Begin capturing packets between the TOE and the remote entity. 3. Cause the TOE to establish a connection to the remote entity. 4. Stop capturing packets between the TOE and the remote entity. 5. Verify the connection is unsuccessful. (e) Setting the path length field of a valid CA field to a value strictly less than the certificate path: 1. Configure the remote entity to present a certificate chain to the TOE such that the issuing certificate of the remote node entity certificate has a path length value strictly less than the certificate path. 2. Begin capturing packets between the TOE and the remote entity. 3. Cause the TOE to establish a connection to the remote entity. 4. Stop capturing packets between the TOE and the remote entity. 5. Verify the connection is unsuccessful. (f1) Valid CA certificates: 1. Ensure the trusted root CA certificate required to establish a trusted chain of trust against the presented certificates is installed into the TOE's CA trust store. 2. Begin capturing packets between the TOE and the remote entity. 3. Cause the TOE to establish a connection to the remote entity. 4. Stop capturing packets between the TOE and the remote entity. 5. Verify the connection is successful. (f2) Removal of trust in one of the CA certificates: 1. Ensure the trusted root CA certificate required to establish a trusted chain of trust against the presented certificates is removed from the TOE's CA trust store. 2. Begin capturing packets between the TOE and the remote entity. 3. Cause the TOE to establish a connection to the remote entity. 4. Stop capturing packets between the TOE and the remote entity. 5. Verify the connection is unsuccessful.
Test Results:	This activity passes as the evaluator observed that the TOE rejects a certificate without a valid certification path resulting in the communications channel not being established, for each of the following reasons: • by establishing a certificate path in which one of the issuing certificates is

	not a CA certificate, • by omitting the basicConstraints field in one of the issuing certificates, • by setting the basicConstraints field in an issuing certificate to have CA=False, • by omitting the CA signing bit of the key usage field in an issuing certificate, and • by setting the path length field of a valid CA field to a value strictly less than the certificate path. When all proper X509 conditions were met the evaluator observed the connection was successfully established.

029	**[APP_PP]FIA_X509_EXT.1.1 – X.509 Certificate Validation**
Test Purpose:	The tests described must be performed in conjunction with the other certificate services evaluation activities, including the functions in FIA_X509_EXT.2.1. The tests for the extendedKeyUsage rules are performed in conjunction with the uses that require those rules. If the application supports chains of length four or greater, the evaluator shall create a chain of at least four certificates: the node certificate to be tested, two Intermediate CAs, and the self-signed Root CA. If the application supports a maximum trust depth of two, then a chain with no Intermediate CA should instead be created. Test 2: The evaluator shall demonstrate that validating an expired certificate results in the function failing.
Test Instructions:	Manually execute this test per the test steps.
Test Procedures:	1. Configure the remote entity to present a certificate chain to the TOE such that the node entity certificate is expired. 2. Begin capturing packets between the TOE and the remote entity. 3. Cause the TOE to establish a connection to the remote entity. 4. Stop capturing packets between the TOE and the remote entity. 5. Verify the connection is unsuccessful.
Test Results:	This activity passes as the evaluator observed that the TOE rejects an expired certificate resulting in the communications channel not being established.

030	**[APP_PP]FIA_X509_EXT.1.1 – X.509 Certificate Validation**
Test Purpose:	The tests described must be performed in conjunction with the other certificate services evaluation activities, including the functions in FIA_X509_EXT.2.1. The tests for the extendedKeyUsage rules are performed in conjunction with the uses that require those rules. If the application supports chains of length four or greater, the evaluator shall create a chain of at least four certificates: the node certificate to be tested, two Intermediate CAs, and the self-signed Root CA. If the application supports a maximum trust depth of two, then a chain with no Intermediate CA should instead be created. Test 3: The evaluator shall test that the TOE can properly handle revoked certificates-"conditional on whether CRL, OCSP, OCSP Stapling or OCSP Multi-stapling is selected; if multiple methods are selected, then the following tests shall be performed for each method: • The evaluator shall test revocation of the node certificate. • The evaluator shall also test revocation of an intermediate CA certificate (i.e. the intermediate CA certificate should be revoked by the root CA), if intermediate CA certificates are supported. If OCSP stapling per RFC

	6066 is the only supported revocation method, this test is omitted. • 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 has been revoked (for each method chosen in the selection) to ensure when the certificate is no longer valid that the validation function fails.
Test Instructions:	Manually execute this test per the test steps.
Test Procedures:	1. Begin capturing packets between the TOE and the remote Exchange server. 2. Cause the TOE to establish a connection to the remote Exchange server. 3. Stop capturing packets between the TOE and the remote Exchange server. 4. Verify the connection is successful to the remote Exchange server. 5. Revoke the remote Exchange server certificate. 6. Repeat Steps 1-3. 7. Verify the connection is unsuccessful to the remote Exchange server. 8. Unrevoke the remote Exchange server certificate. 9. Revoke the intermediate01 CA certificate. 10. Repeat Steps 1-3. 11. Verify the connection is unsuccessful to the remote Exchange server.
Test Results:	This activity passes as the evaluator observed that for all revoked conditions, the TOE properly identified and connections were not established. All non-revoked conditions were properly identified and the connection was established.

031	**[APP_PP]FIA_X509_EXT.1.1 – X.509 Certificate Validation – TD0780**
Test Purpose:	The tests described must be performed in conjunction with the other certificate services evaluation activities, including the functions in FIA_X509_EXT.2.1. The tests for the extendedKeyUsage rules are performed in conjunction with the uses that require those rules. If the application supports chains of length four or greater, the evaluator shall create a chain of at least four certificates: the node certificate to be tested, two Intermediate CAs, and the self-signed Root CA. If the application supports a maximum trust depth of two, then a chain with no Intermediate CA should instead be created. Test 4: If any OCSP option is selected, the evaluator shall configure the TSF to reject certificates if it cannot access valid status information, if so configurable. Then the evaluator shall ensure the TSF has no other source of revocation information available and configure the OCSP server or use a man-in-the-middle tool to present an OCSP response signed by a certificate that does not have the OCSP signing purpose and which is the only source of revocation status information advertised by the CA issuing the certificate being validated. The evaluator shall verify that validation of the OCSP response fails and that the TOE treats the certificate being checked as invalid and rejects the connection. If CRL is selected, the evaluator shall likewise configure the CA to be the only source of revocation status information, and sign a CRL with a certificate that does not have the cRLsign key usage bit set. The evaluator shall verify that validation of the CRL fails and that the TOE treats the certificate being checked as invalid and rejects the connection.
Test Instructions:	Manually execute this test per the test steps.
Test Procedures:	1. Begin capturing packets between the TOE and the remote Exchange server. 2. Cause the TOE to establish a connection to the remote Exchange server

	and present a OCSP signing certificate to the TOE that does not have the OCSP signing purpose. 3. Stop capturing packets between the TOE and the remote Exchange server. 4. Verify the connection is unsuccessful to the remote Exchange server.
Test Results:	This activity passes as the evaluator observed the TOE properly identified that the OCSP singing purpose was missing in the OCSP signing certificate and the connection was not established.

032	**[APP_PP]FIA_X509_EXT.1.1 – X.509 Certificate Validation**
Test Purpose:	The tests described must be performed in conjunction with the other certificate services evaluation activities, including the functions in FIA_X509_EXT.2.1. The tests for the extendedKeyUsage rules are performed in conjunction with the uses that require those rules. If the application supports chains of length four or greater, the evaluator shall create a chain of at least four certificates: the node certificate to be tested, two Intermediate CAs, and the self-signed Root CA. If the application supports a maximum trust depth of two, then a chain with no Intermediate CA should instead be created. Test 5: The evaluator shall modify any byte in the first eight bytes of the certificate and demonstrate that the certificate fails to validate. (The certificate will fail to parse correctly.)
Test Instructions:	Manually execute this test per the test steps.
Test Procedures:	1. Begin capturing packets between the TOE and the remote entity. 2. Cause the TOE to establish a connection to the remote entity and using a man-in-the-middle-tool, modify any byte in the first eight bytes of the node certificate. 3. Stop capturing packets between the TOE and the remote entity. 4. Verify the connection is unsuccessful.
Test Results:	This activity passes as the evaluator observed the TOE properly identified that the OCSP singing purpose was missing and connection was not established.

033	**[APP_PP]FIA_X509_EXT.1.1 – X.509 Certificate Validation**
Test Purpose:	The tests described must be performed in conjunction with the other certificate services evaluation activities, including the functions in FIA_X509_EXT.2.1. The tests for the extendedKeyUsage rules are performed in conjunction with the uses that require those rules. If the application supports chains of length four or greater, the evaluator shall create a chain of at least four certificates: the node certificate to be tested, two Intermediate CAs, and the self-signed Root CA. If the application supports a maximum trust depth of two, then a chain with no Intermediate CA should instead be created. Test 6: The evaluator shall modify any byte in the last byte of the certificate and demonstrate that the certificate fails to validate. (The signature on the certificate will not validate.)
Test Instructions:	Manually execute this test per the test steps.
Test Procedures:	1. Begin capturing packets between the TOE and the remote entity. 2. Cause the TOE to establish a connection to the remote entity and using a man-in-the-middle-tool, modify any byte in the last byte of the node certificate. 3. Stop capturing packets between the TOE and the remote entity.

	4. Verify the connection is unsuccessful.
Test Results:	This activity passes as the evaluator observed the TOE properly identified that the certificate was modified and the connection was not established.

034	**[APP_PP]FIA_X509_EXT.1.1 – X.509 Certificate Validation**
Test Purpose:	The tests described must be performed in conjunction with the other certificate services evaluation activities, including the functions in FIA_X509_EXT.2.1. The tests for the extendedKeyUsage rules are performed in conjunction with the uses that require those rules. If the application supports chains of length four or greater, the evaluator shall create a chain of at least four certificates: the node certificate to be tested, two Intermediate CAs, and the self-signed Root CA. If the application supports a maximum trust depth of two, then a chain with no Intermediate CA should instead be created. Test 7: The evaluator shall modify any byte in the public key of the certificate and demonstrate that the certificate fails to validate. (The signature on the certificate will not validate.)
Test Instructions:	Manually execute this test per the test steps.
Test Procedures:	1. Begin capturing packets between the TOE and the remote entity. 2. Cause the TOE to establish a connection to the remote entity and using a man-in-the-middle-tool, modify any byte in the public key portion of the node certificate. 3. Stop capturing packets between the TOE and the remote entity. 4. Verify the connection is unsuccessful.
Test Results:	This activity passes as the evaluator observed the TOE properly identified that the certificate was modified and the connection was not established.

035	**[APP_PP]FIA_X509_EXT.1.1 – X.509 Certificate Validation**
Test Purpose:	The tests described must be performed in conjunction with the other certificate services evaluation activities, including the functions in FIA_X509_EXT.2.1. The tests for the extendedKeyUsage rules are performed in conjunction with the uses that require those rules. If the application supports chains of length four or greater, the evaluator shall create a chain of at least four certificates: the node certificate to be tested, two Intermediate CAs, and the self-signed Root CA. If the application supports a maximum trust depth of two, then a chain with no Intermediate CA should instead be created. Test 8: (Conditional on support for EC certificates as indicated in FCS_COP.1/Sig). The evaluator shall establish a valid, trusted certificate chain consisting of an EC leaf certificate, an EC Intermediate CA certificate not designated as a trust anchor, and an EC certificate designated as a trusted anchor, where the elliptic curve parameters are specified as a named curve. The evaluator shall confirm that the TOE validates the certificate chain.
Test Instructions:	Manually execute this test per the test steps.
Test Procedures:	*NOTE: Per the test assurance activity, this test is conditional upon support for EC certificates as indicated in FCS_COP.1/Sig). The Security Target, for FCS_COP.1/Sig, does not define support for EC certificates. Therefore, this conditional test does not apply.*
Test Results:	This activity is considered satisfied.

036	[APP_PP]FIA_X509_EXT.1.1 – X.509 Certificate Validation
Test Purpose:	The tests described must be performed in conjunction with the other certificate services evaluation activities, including the functions in FIA_X509_EXT.2.1. The tests for the extendedKeyUsage rules are performed in conjunction with the uses that require those rules. If the application supports chains of length four or greater, the evaluator shall create a chain of at least four certificates: the node certificate to be tested, two Intermediate CAs, and the self-signed Root CA. If the application supports a maximum trust depth of two, then a chain with no Intermediate CA should instead be created. Test 9: (Conditional on support for EC certificates as indicated in FCS_COP.1/Sig). The evaluator shall replace the intermediate certificate in the certificate chain for Test 8a with a modified certificate, where the modified intermediate CA has a public key information field where the EC parameters uses an explicit format version of the Elliptic Curve parameters in the public key information field of the intermediate CA certificate from Test 8a, and the modified Intermediate CA certificate is signed by the trusted EC root CA, but having no other changes. The evaluator shall confirm the TOE treats the certificate as invalid.
Test Instructions:	Manually execute this test per the test steps.
Test Procedures:	*NOTE: Per the test assurance activity, this test is conditional upon support for EC certificates as indicated in FCS_COP.1/Sig. The Security Target, for FCS_COP.1/Sig, does not define support for EC certificates. Therefore, this conditional test does not apply.*
Test Results:	This activity is considered satisfied.

037	[APP_PP]FIA_X509_EXT.1.2 – X.509 Certificate Validation
Test Purpose:	The tests described must be performed in conjunction with the other certificate services evaluation activities, including the functions in FIA_X509_EXT.2.1. If the application supports chains of length four or greater, the evaluator shall create a chain of at least four certificates: the node certificate to be tested, two Intermediate CAs, and the self-signed Root CA. If the application supports a maximum trust depth of two, then a chain with no Intermediate CA should instead be created. Test 1: The evaluator shall ensure that the certificate of at least one of the CAs in the chain does not contain the basicConstraints extension. The evaluator shall confirm that validation of the certificate path fails (i) as part of the validation of the peer certificate belonging to this chain; and/or (ii) when attempting to add the CA certificate without the basicConstraints extension to the TOE's trust store.
Test Instructions:	Manually execute this test per the test steps.
Test Procedures:	1. Configure the remote entity to present to the TOE a certificate chain such that the certificate of the CA issuing the remote node entity certificate does not contain the basicConstraints extension. 2. Begin capturing packets between the TOE and the remote entity. 3. Cause the TOE to establish a connection to the remote entity. 4. Stop capturing packets between the TOE and the remote entity. 5. Verify the connection is unsuccessful.
Test Results:	This activity passes as the evaluator observed the TOE properly identified that the certificate was missing the basicConstraints extension in the issuing CA certificate and the connection was not established.

038	[APP_PP]FIA_X509_EXT.1.2 – X.509 Certificate Validation
Test Purpose:	The tests described must be performed in conjunction with the other certificate services evaluation activities, including the functions in FIA_X509_EXT.2.1. If the application supports chains of length four or greater, the evaluator shall create a chain of at least four certificates: the node certificate to be tested, two Intermediate CAs, and the self-signed Root CA. If the application supports a maximum trust depth of two, then a chain with no Intermediate CA should instead be created. Test 2: The evaluator shall ensure that the certificate of at least one of the CAs in the chain has the CA flag in the basicConstraints extension not set (or set to FALSE). The evaluator shall confirm that validation of the certificate path fails (i) as part of the validation of the peer certificate belonging to this chain; and/or (ii) when attempting to add the CA certificate with the CA flag not set (or set to FALSE) in the basicConstraints extension to the TOE's trust store.
Test Instructions:	Manually execute this test per the test steps.
Test Procedures:	1. Configure the remote entity to present to the TOE a certificate chain such that the certificate of the CA issuing the remote node entity certificate has the CA flag not set (i.e. CA=False) in the basicConstraints extension. 2. Begin capturing packets between the TOE and the remote entity. 3. Cause the TOE to establish a connection to the remote entity. 4. Stop capturing packets between the TOE and the remote entity. 5. Verify the connection is unsuccessful.
Test Results:	This activity passes as the evaluator observed the TOE properly identified that the issuing CA certificate had the CA flag not set and the connection was not established.

4.3.3.2 FIA_X509_EXT.2

039	[APP_PP]FIA_X509_EXT.2.2 – X.509 Certificate Authentication
Test Purpose:	The evaluator shall perform the following test for each trusted channel: 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.
Test Instructions:	Manually execute this test per the test steps.
Test Procedures:	Admin Configure: Always Reject if OCSP Unavailable Configure the TOE such that “PolicyTLSRevocationStrictness” is set to “Strict” (PolicyTLSRevocationStrictness == 2). 1. Begin capturing packets between the TOE and the Exchange server. 2. Perform some action on the TOE that causes it to perform a check of the validity of the Exchange server certificate. 3. (optional): Stop capturing packets between the TOE and the Exchange server. 4. Verify that the TOE successfully verified the validity of the Exchange server certificate, and that the TLS connection to the Exchange server was successful.

	5. Manipulate the environment such that the TOE is unable to verify the validity of the Exchange server certificate. 6. Repeat Steps 2-4. 7. Verify that the TOE was unable to verify the validity of the Exchange server certificate and failed to establish a TLS connection to the Exchange server. 8. Reconfigure the environment such that the TOE can verify the validity of the Exchange server certificate. Admin Configure: Accept if last known response is valid 9. Configure the TOE such that "PolicyTLSRevocationStrictness" is set to "Moderate" (PolicyTLSRevocationStrictness == 1). 10. Begin capturing packets between the TOE and the Exchange server. 11. Perform some action on the TOE that causes it to perform a check of the validity of the Exchange server certificate. 12. (optional): Stop capturing packets between the TOE and the Exchange server. 13. Verify that the TOE successfully verified the validity of the Exchange server certificate, and that the TLS connection to the Exchange server was successful. 14. Manipulate the environment such that the TOE is unable to verify the validity of the Exchange server certificate. 15. Repeat Steps 11-13 16. Verify that the TOE was able to verify the validity of the Exchange server certificate without a response from the OCSP responder and successfully established a TLS connection to the Exchange server. 17. Reconfigure the environment such that the TOE can verify the validity of the Exchange server certificate. Admin Configure: Reject if last known response is unknown 18. Begin capturing packets between the TOE and the Exchange server. 19. Perform some action on the TOE that causes it to perform a check of the validity of the Exchange server certificate. 20. (optional): Stop capturing packets between the TOE and the Exchange server. 21. Verify that the TOE successfully verified the validity of the Exchange server certificate, and that the TLS connection to the Exchange server was successful. 22. Manipulate the environment such that the TOE is unable to verify the validity of the Exchange server certificate. 23. Repeat Steps 19-21. 24. Verify that the TOE was unable to verify the validity of the Exchange server certificate without a response from the OCSP responder and failed to establish a TLS connection to the Exchange server. 25. Reconfigure the environment such that the TOE can verify the validity of the Exchange server certificate.

	Admin Configure: Reject if last known is revoked 26. Revoke the Exchange server certificate. 27. Begin capturing packets between the TOE and the Exchange server. 28. Perform some action on the TOE that causes it to perform a check of the validity of the Exchange server certificate. 29. (optional): Stop capturing packets between the TOE and the Exchange server. 30. Verify that the TOE successfully verified the validity of the Exchange server certificate, and that the TLS connection to the Exchange server was unsuccessful. 31. Manipulate the environment such that the TOE is unable to verify the validity of the Exchange server certificate. 32. Repeat Steps 28-30. 33. Verify that the TOE was unable to verify the validity of the Exchange server certificate without a response from the OCSP responder and failed to establish a TLS connection to the Exchange server. 34. Reconfigure the environment such that the TOE can verify the validity of the Exchange server certificate.
Test Results:	This activity passes as the evaluator observed the TOE: • correctly rejected the certificate and did not establish the connection when the TSF was set to always reject when the OCSP is unavailable: • correctly rejected the certificate and did not establish the connection when the TSF was set to use previous known value when the OCSP is unavailable and the previous value was revoked. • correctly rejected the certificate and did not establish the connection when the TSF was set to use previous known value when the OCSP is unavailable and the previous value was unknown. • correctly accepted the certificate and did establish the connection when the TSF was set to use previous known value when the OCSP is unavailable and the previous value was valid.

040	**[APP_PP]FIA_X509_EXT.2.2 – X.509 Certificate Authentication**
Test Purpose:	The evaluator shall perform the following test for each trusted channel: Test 2: The evaluator shall demonstrate that an invalid certificate that requires certificate validation checking to be performed in at least some part by communicating with a non-TOE IT entity cannot be accepted.
Test Instructions:	Manually execute this test per the test steps.
Test Procedures:	1. Configure the remote entity to present a certificate chain to the TOE such that the Exchange server certificate is expired and contains revocation check location information in the AIA field of the certificate. 2. Begin capturing packets between the TOE and the Exchange server. 3. Cause the TOE to establish a connection to the remote Exchange server. 4. Stop capturing packets between the TOE and the remote Exchange server.

	5. Verify the connection is unsuccessful to the remote Exchange server.
Test Results:	This activity passes as the evaluator observed that the TOE correctly did not accept an expired (a type of invalid) certificate, containing a revocation status URI reference, that was presented to TOE as part of a TLS authentication and did not establish the connection.

4.3.3.3 FIA_X509_EXT.3

041	**[EC_PP]FIA_X509_EXT.3.1 – X.509 Authentication and Encryption**
Test Purpose:	The evaluator shall perform the following tests: Test 1: The evaluator shall perform Test 1 for each function listed in FIA_X509_EXT.2.1 ([AppPP]) that requires the use of certificates. The evaluator shall demonstrate that using a certificate without a valid certification path results in the function failing. The evaluator shall then load into the platform's root store any certificates needed to validate the certificate to be used in the function, and demonstrate that the function succeeds.
Test Instructions:	Manually execute this test per the test steps.
Test Procedures:	1. Ensure all CA certificates required to validate the S/MIME signature are present in the TOE CA trust store. 2. Send a signed email to the TOE. 3. Verify that the TOE successfully validates the S/MIME signature. 4. Remove the Intermediate 01 CA certificate from the TOE CA trust store. Verify that the TOE fails to validate the S/MIME signature.
Test Results:	This activity passes as the evaluator observed the TOE successfully validated the S/MIME signature when the CA certification chain required to validate the signature was present in the TOE's trust store. Additionally, it was observed that the TOE could not validate the S/MIME signature when one of the intermediate CA certificates was removed from the TOE's trust store.

042	**[EC_PP]FIA_X509_EXT.3.1 – X.509 Authentication and Encryption**
Test Purpose:	The evaluator shall perform the following tests: 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 email client is unable to verify the validity of the certificate, and observe that the action selected in FIA_X509_EXT.2.2 ([AppPP]) 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 Instructions:	Manually execute this test per the test steps.
Test Procedures:	Admin Configure: Always Reject if OCSP Unavailable 1. Configure the TOE such that “PolicySMIMERevocationStrictness” is set to “Strict” (PolicySMIMERevocationStrictness == 2). 2. Begin capturing packets between the TOE and the OCSP responder. 3. Perform some action on the TOE that causes it to perform an OCSP check. 4. (optional first round): Stop capturing packets between the TOE and the OCSP responder. 5. Verify that the TOE successfully communicated with the OCSP

	responder, and that the signature of the e-mail was successfully validated. 6. Disconnect the connection between the TOE and the OCSP responder. 7. Repeat Steps 2-4. 8. Verify that the TOE attempted to communicate with the OCSP responder and failed to validate the signature of the e-mail. 9. Reconnect the connection between the TOE and the OCSP responder. Admin Configure: Reject if last known response is unknown 10. Begin capturing packets between the TOE and the OCSP responder. 11. Perform some action on the TOE that causes it to perform an OCSP check. 12. (optional first round): Stop capturing packets between the TOE and the OCSP responder. 13. Verify that the TOE successfully communicated with the OCSP responder, and that the signature of the e-mail was successfully validated. 14. Disconnect the connection between the TOE and the OCSP responder. 15. Uninstall the TOE application. (required in order to clear cache) 16. Re-install the TOE application. 17. Repeat Steps 10-12. 18. Verify that the TOE attempted to communicate with the OCSP responder and failed to validate the signature of the e-mail. 19. Reconnect the connection between the TOE and the OCSP responder.
Test Results:	This activity passes as the evaluator observed the TOE: • correctly rejected the certificate and did not establish the connection when the TSF was set to always reject when the OCSP is unavailable: • correctly rejected the certificate and did not establish the connection when the TSF was set to use previous known value when the OCSP is unavailable and the previous value was unknown. The condition of the status value in the cache is “known” or “revoked” will not reach out to the OCSP but will make the decision based on the status value.

4.3.4 Security Management

4.3.4.1 FMT_CFG_EXT.1

043	**[APP_PP]FMT_CFG_EXT.1.1 – Secure by Default Configuration – Test 1**
Test Purpose:	If the application uses any default credentials the evaluator shall run the following tests. Test 1: The evaluator shall install and run the application without generating or loading new credentials and verify that only the minimal application functionality required to set new credentials is available.
Test Instructions:	Manually execute this test per the test steps.
Test Procedures:	*N/A - There are no default credentials for the TOE.*
Test Results:	This activity is considered satisfied.

044	**[APP_PP]FMT_CFG_EXT.1.1 – Secure by Default Configuration – Test 2**
Test Purpose:	If the application uses any default credentials the evaluator shall run the following tests. Test 2: The evaluator shall attempt to clear all credentials and verify that only the minimal application functionality required to set new credentials is available.
Test Instructions:	Manually execute this test per the test steps.
Test Procedures:	*N/A - There are no default credentials for the TOE.*
Test Results:	This activity is considered satisfied.

045	**[APP_PP]FMT_CFG_EXT.1.1 – Secure by Default Configuration – Test 3**
Test Purpose:	If the application uses any default credentials the evaluator shall run the following tests. Test 3: The evaluator shall run the application, establish new credentials and verify that the original default credentials no longer provide access to the application.
Test Instructions:	Manually execute this test per the test steps.
Test Procedures:	*N/A - There are no default credentials for the TOE.*
Test Results:	This activity is considered satisfied.

046	**[APP_PP]FMT_CFG_EXT.1.2 – Secure by Default Configuration**

Test Purpose:	The evaluator shall install and run the application. The evaluator shall inspect the filesystem of the platform (to the extent possible) for any files created by the application and ensure that their permissions are adequate to protect them. The method of doing so varies per platform. Platforms: Android... The evaluator shall run the command find -L . -perm /002 inside the application's data directories to ensure that all files are not world-writable. The command should not print any files. platforms: Apple iOS... The evaluator shall determine whether the application leverages the appropriate Data Protection Class for each data file stored locally.
Test Instructions:	Manually execute this test per the test steps.
Test Procedures:	**Android:** 1. Install and run the TOE application. 2. Using adb (NOTE: if on non-rooted device, need to execute with debug build using adb run-as com.boxer.email), run the following command inside the TOE's application data directories (/data/user/0/com.boxer.email and /data/data/com.boxer.email): find -L . -perm /002 3. Verify that all files are not world-writeable. 4. Inspect external storage and verify that no sensitive data is written to it. **iOS / iPadOS:** 1. Verify that the TOE application leverages the appropriate Data Protection Class for each data file stored locally. a. Refer to FDP_DAR_EXT.1.1 – Test Case 017 – iOS.
Test Results:	This activity passes as the evaluator observed that there were no world writable files, no sensitive data written to external storage devices on the Android device. The iOS / iPad devices inherently meet the requirement with the mandatory enforcement of application data being written to the sandbox.

4.3.4.2 FMT_MEC_EXT.1

047	**[APP_PP]FMT_MEC_EXT.1.1 – Supported Configuration Mechanism – TD0747**
Test Purpose:	If "invoke the mechanisms recommended by the platform vendor for storing and setting configuration options" is chosen, the method of testing varies per platform as follows: Platforms: Android... The evaluator shall inspect the TSS and verify that it describes what Android API is used (and provides a link to the documentation of the API) when storing configuration data. The evaluator shall run the application and verify that the behavior of the TOE is consistent with where and how the API documentation says

	the configuration data will be stored. Platforms: Apple iOS... The evaluator shall verify that the app uses the user defaults system or key-value store for storing all settings.
Test Instructions:	Manually execute this test per the test steps.
Test Procedures:	**Android:** 1. Using adb (NOTE: if on non-rooted device, need to execute with debug build using adb run-as com.boxer.email), capture the state of the XML files located in: /data/data/com.boxer.email/shared_prefs 2. Run the TOE application. 3. Make security-related configuration changes to the TOE: a. Change the TOE application passcode. 4. Repeat Step 1. 5. Compare the states of the XML files and verify that at least one of the XML files reflects the changes made to the TOE configuration: a. Check for changes to the value for key: "~κλειδί~" in /data/data/com.boxer.email/shared_prefs/com.boxer.email_preferences.xml 6. Verify that the TOE application utilized the SharedPreferences and/or PreferenceActivity classes, per API documentation. **iOS/iPadOS:** 1. Verify that the TOE application uses the user defaults system or key-value store for storing all settings by examining the source code.
Test Results:	This activity passes as the evaluator observed the TOE used the Android API to store the configuration data (TOE application passcode) is the Android SharedPrefs: https://developer.android.com/reference/android/content/SharedPreferences. Additionally, it was observed that the iOS/iPadOS stored keys in the iOS key-value store.

4.3.4.3 FMT_MOF_EXT.1

048	**[EC_EP]FMT_MOF_EXT.1.1 – Supported Configuration Mechanism**
Test Purpose:	The evaluator shall perform the following tests: Test 1: The evaluator shall verify that functions perform as intended by enabling, disabling, and configuring the functions.
Test Instructions:	Manually execute this test per the test steps.
Test Procedures:	Enable/disable plaintext only mode globally 1. Enable plaintext only mode globally (Refer to AGD supplement Section 7.2). 2. Send an e-mail containing HTML content to the TOE. 3. Verify that the e-mail is viewed in plaintext. 4. Disable plaintext only mode globally. 5. Send an e-mail containing HTML content to the TOE. 6. Verify that the HTML content in the e-mail is rendered. Configure message sending/receiving to only use cryptographic algorithms defined in FCS_SMIME_EXT.1

	NOTE: This is performed during initial setup as part of placing the TOE in the evaluated configuration. (Refer to AGD supplement Section 6.2.2). Change password/passphrase authentication credential Android: a. Launch the TOE application. b. Authenticate to the TOE. c. Tap "Settings" > "Change shared passcode". d. Enter the current passcode. e. Specify the new passcode. f. Verify the new passcode. iOS / iPadOS: a. Launch the TOE application. b. Authenticate to the TOE. c. Tap "Settings" > "Advanced" > "Passcode" > "Change Single Sign-On passcode". d. Enter the current passcode. e. Specify the new passcode. f. Verify the new passcode. Configure cryptographic functionality NOTE: This is performed during initial setup as part of placing the TOE in the evaluated configuration and the configuration exercised in [EC_PP]FCS_SMIME_EXT.1 – Test Cases 008 and 009. (Refer to AGD supplement Section 6.2.2). Configure Password Complexity Policy: Length 1. Configure the TOE password length to 5 complexity policy via UEM. (Refer to AGD supplement Section 7.2.1). 2. Attempt to assign a 4-character password on the TOE: 1234 3. Verify the new password is rejected. 4. Attempt to assign a 5-character password on the TOE: 12345 5. Verify new password is accepted. Configure OCSP retrieval frequency

	NOTE: This is tested in [EC_PP]FCS_SMIME_EXT.1 – Test Case 015 and [EC_EP]FIA_X509_EXT.3 – Test Case 042. (Refer to AGD supplement Section 6.3.1).
Test Results:	This activity passes as the evaluator observed the TOE management functions behaved and were enforced as expected.

049	**[EC_EP]FMT_MOF_EXT.1.1 – Supported Configuration Mechanism**
Test Purpose:	The evaluator shall perform the following tests: Test 2: The evaluator shall set management functions which are controlled by the (enterprise) administrator and cannot be overridden by the user. The evaluator shall apply these functions to the client, attempt to override each setting as the user, and ensure that the email client does not permit it.
Test Instructions:	Manually execute this test per the test steps.
Test Procedures:	Enable/disable plaintext only mode globally 1. Configure the TOE via UEM to enable plaintext only mode globally. (Refer to AGD supplement Section 7.2). 2. On the TOE, attempt to disable plaintext only mode globally. 3. Verify that the TOE fails to disable plaintext only mode globally. Configure message sending/receiving to only use cryptographic algorithms defined in FCS_SMIME_EXT.1 1. Configure the TOE via UEM to only use the cryptographic algorithms defined in FCS_SMIME_EXT.1. (Refer to AGD supplement Section 6.2.2). 2. On the TOE, attempt to enable cryptographic algorithms other than those defined by the UEM. 3. Verify that the TOE fails to enable any additional cryptographic algorithms. Configure cryptographic functionality 1. Configure the TOE's cryptographic functionality via UEM. (Refer to AGD supplement Section 6.2.2). 2. On the TOE, attempt to modify the cryptographic functionality. 3. Verify that the TOE fails to modify the cryptographic functionality that was set via UEM. Configure Password Complexity Policy: Length 1. Configure the TOE password length complexity policy via UEM. (Refer to AGD supplement Section 7.2.1). 2. On the TOE, attempt to modify the password length complexity policy. 3. Verify that the TOE fails to modify the password length complexity policy.

	Configure OCSP retrieval frequency 1. Configure the TOE OCSP retrieval frequency via UEM. (Refer to AGD supplement Section 6.3.1). 2. On the TOE, attempt to modify the OCSP retrieval frequency. 3. Verify that the TOE fails to modify the OCSP retrieval frequency.
Test Results:	This activity passes as the evaluator observed the none of the TOE management functions were accessible by the TOE end user on the mobile platforms.

050	**[EC_EP]FMT_MOF_EXT.1.1 – Supported Configuration Mechanism**
Test Purpose:	The evaluator shall perform the following tests: Test 3: Disable Key Recovery: If the email client provides key recovery capability, then the evaluator shall devise a test that ensures that the key recovery capability has been or can be disabled following the guidance provided by the vendor.
Test Instructions:	Manually execute this test per the test steps.
Test Procedures:	**NOTE:** The key recovery (key escrow) functionality is disabled by default upon installation for CC compliance. The below steps have been provided by the Vendor to strictly verify this feature is disabled and should not be used to enable this feature as it will be outside of the evaluated configuration. **Android:** 1. Via the enterprise UEM console, ensure “Disable Key Escrow (Forgot Passcode)” is set to “YES”. 2. Verify the “Forgot Passcode?” function is disabled. **iOS/iPadOS:** 1. Via the enterprise UEM console, ensure “Disable Key Escrow (Forgot Passcode)” is set to “YES” and “Swift SDK Key Wrapping Only Mode” is set to “ENABLE”. 2. Verify the “Forgot Passcode?” function is disabled.
Test Results:	This activity passes as the evaluator observed that when the key recovery/escrow feature is configured to be disabled by the Administrator from within the UEM, the "Forgot Passcode" function at the TOE passcode screen is disabled. For Android, the "Forgot Passcode" UI element doesn't appear at all when disabled.

4.3.4.4 FMT_SMF.1

051	**[APP_PP]FMT_SMF.1.1 – Specification of Management Functions**
Test Purpose:	The evaluator shall test the application's ability to provide the management functions by configuring the application and testing each option selected from above. The evaluator is expected to test these functions in all the ways in which the ST and guidance documentation state the configuration can be managed.
Test Instructions:	Manually execute this test per the test steps.
Test Procedures:	NOTE: This is tested in [EC_EP]FMT_MOF_EXT.1 – Test Case 048.
Test Results:	This activity is considered satisfied by test case 048.

4.3.5 Privacy

4.3.5.1 FPR_ANO_EXT.1

052	**[APP_PP]FPR_ANO_EXT.1.1 – User Consent for Transmission of Personally Identifiable Information**
Test Purpose:	If require user approval before executing is selected, the evaluator shall run the application and exercise the functionality responsibly for transmitting PII and verify that user approval is required before transmission of the PII.
Test Instructions:	Manually execute this test per the test steps.
Test Procedures:	*NOTE: Per the Security Target, "The TOE application does not collect personally identifiable information (PII) for administrators or users. Therefore, the TOE application will not transmit PII data over the network unless the owner of the mobile device includes such information in the free text email. Free text in an email is outside the TOE's scope of control."*
Test Results:	This activity is considered satisfied.

4.3.6 Protection of the TSF

4.3.6.1 FPT_AEX_EXT.1

053	**[APP_PP]FPT_AEX_EXT.1.1 – Anti-Exploitation Capabilities – TD0798**
Test Purpose:	The evaluator shall perform either a static or dynamic analysis to determine that no memory mappings are placed at an explicit and consistent address except for any exceptions claimed in the SFR. For these exceptions, the evaluator shall verify that this analysis shows explicit mappings that are consistent with what is claimed in the TSS. The method of doing so varies per platform. For those platforms requiring the same application running on two different systems, the evaluator may alternatively use the same device. After collecting the first instance of mappings, the evaluator must uninstall the application, reboot the device, and reinstall the application to collect the second instance of mappings. Platforms: Android... The evaluator shall run the same application on two different Android systems. Both devices do not need to be evaluated, as the second device is acting only as a tool. Connect via ADB and inspect /proc/PID/maps. Ensure the two different instances share no memory mappings made by the application at the same location. platforms: Apple iOS... The evaluator shall perform a static analysis to search for any mmap calls (or API calls that call mmap), and ensure that no arguments are provided that request a mapping at a fixed address.
Test Instructions:	Manually execute this test per the test steps.
Test Procedures:	Android: 1. Run the TOE application. 2. Execute "ps -A | grep com.boxer.email" to identify the PID of the TOE. 3. Connect via ADB and inspect /proc/<PID>/maps 4. On the same Android device, uninstall the TOE application.

	5. Reboot the Android device. 6. Install the TOE application on the Android device. 7. Repeat Steps 1 – 3. 8. Verify that the two different instances share no mapping locations. iOS / iPadOS: 1. Perform a static code analysis of the TOE's source code by searching for any mmap calls, including API calls that call mmap. 2. For any mmap calls, verify that no arguments are provided that request a mapping at a fixed address.
Test Results:	This activity passes as the evaluator observed on the Android platform, there were no memory mapping spaces that were the same between the two instances that were in control of the TOE. For the iOS/iPadOS platform a search for mmap calls resulted in no arguments for requesting a mapping to a fixed address.

054	**[APP_PP]FPT_AEX_EXT.1.2 – Anti-Exploitation Capabilities**
Test Purpose:	The evaluator shall verify that no memory mapping requests are made with write and execute permissions. The method of doing so varies per platform. Platforms: Android... The evaluator shall perform static analysis on the application to verify that mmap is never invoked with both the PROT_WRITE and PROT_EXEC permissions, and mprotect is never invoked. Platforms: Apple iOS... The evaluator shall perform static analysis on the application to verify that mprotect is never invoked with the PROT_EXEC permission.
Test Instructions:	Manually execute this test per the test steps.
Test Procedures:	Android: 1. Perform a static code analysis of the TOE's source code by searching for any mmap calls. 2. For any mmap calls, verify that it is never invoked with both the PROT_WRITE and PROT_EXEC permissions. 3. Verify that mprotect is never invoked. iOS/iPadOS: 1. Perform a static code analysis of the TOE's source code by searching for any mprotect calls. 2. For any mprotect calls, verify that it is never invoked with the PROT_EXEC permission.
Test Results:	This activity passes as the evaluator observed during an Android Boxer code review that PROT_WRITE and PROT_EXEC are never invoked together and that mprotect is never invoked. Additionally, for the iOS/iPadOS code review it was found that mprotect is never invoked.

055	[APP_PP]FPT_AEX_EXT.1.3 – Anti-Exploitation Capabilities
Test Purpose:	The evaluator shall configure the platform in the ascribed manner and carry out one of the prescribed tests: Platforms: Android... Applications running on Android cannot disable Android security features, therefore this requirement is met and no evaluation activity is required. platforms: Apple iOS... Applications running on iOS cannot disable security features, therefore this requirement is met and no evaluation activity is required.
Test Instructions:	Manually execute this test per the test steps.
Test Procedures:	*NOTE: Per the test assurance activity, this requirement is met, and no evaluation activity is required.*
Test Results:	This activity is considered satisfied.

056	[APP_PP]FPT_AEX_EXT.1.4 – Anti-Exploitation Capabilities
Test Purpose:	The evaluator shall run the application and determine where it writes its files. For files where the user does not choose the destination, the evaluator shall check whether the destination directory contains executable files. This varies per platform: Platforms: Android... The evaluator shall run the program, mimicking normal usage, and note where all user-modifiable files are written. The evaluator shall ensure that there are no executable files stored under /data/data/package/ where package is the Java package of the application. Platforms: Apple iOS... The evaluator shall consider the requirement met because the platform forces applications to write all data within the application working directory (sandbox).
Test Instructions:	Manually execute this test per the test steps.
Test Procedures:	Android: 1. Run the TOE application. 2. Perform normal functions/usage of the application (e.g., send an encrypted and signed e-mail, receive an encrypted and signed e-mail, etc.) 3. Notate where all user-modifiable files are written. 4. Verify that there are no executable files stored under /data/data/com.boxer.email 5. Verify the files notated in Step 3 are not executable. iOS / iPadOS: N/A – Per test assurance activity, this requirement is automatically met because the platform forces applications to write all data within the application working directory (sandbox).
Test Results:	This activity passes as the evaluator observed on the Android platform, there were no user-modifiable files, controlled by the TOE, written to the /data/data/com.boxer.email directory. The iOS/iPadOS platform had no assurance activity to perform.

057	[APP_PP]FPT_AEX_EXT.1.5 - TD0815 – Anti-Exploitation Capabilities
Test Purpose:	The evaluator will inspect every native executable included in the TOE to ensure that stack-based buffer overflow protection is present. For PE , the evaluator will disassemble each and ensure the following sequence appears: mov rcx, QWORD PTR [rsp+(...)] xor rcx, (...) call (...) . For ELF executables, the evaluator will ensure that each contains references to the symbol __stack_chk_fail. Tools such as Canary Detector may help automate these activities. If these automated tests fail, the evaluator shall perform the above, conditional TSS activity.
Test Instructions:	Manually execute this test per the test steps.
Test Procedures:	Android: Inspect the compilation procedures and/or make files to verify that the “-fstack-protector-strong” or “-fstack-protector-all” flags are used. iOS / iPadOS: If the TOE application is compiled using GCC or Xcode, inspect the compilation procedures to verify that the “-fstack-protector-strong” or “-fstack-protector-all” flags are used. If the TOE application is compiled using any other compiler, verify that appropriate stack-protection mechanisms were used during the build process.
Test Results:	This activity passes as the evaluator reviewed and witnessed appropriate stack-protection mechanisms are used during the build process, fstack-protector-strong for Android and fstack-protector-all for iOS/iPadOS platforms.

4.3.6.2 FPT_AON_EXT.1

058	[EC_EP]FPT_AON_EXT.1.1 – Support for Only Trusted Add-ons
Test Purpose:	The evaluator shall perform the following test: Test 1: The evaluator shall create or obtain an untrusted add-on and attempt to load it. The evaluator shall verify that the untrusted add-on is rejected and cannot be loaded.
Test Instructions:	Manually execute this test per the test steps.
Test Procedures:	N/A – The TOE does not support the installation of trusted or untrusted add-ons.
Test Results:	This activity is considered satisfied.

4.3.6.3 FPT_API_EXT.1

059	[EC_EP]FPT_API_EXT.1.1 – Use of Supported Services and APIs
Test Purpose:	The evaluator shall then compare the list with the supported APIs (available through e.g. developer accounts, platform developer groups) and ensure that all APIs listed in the TSS are supported.
Test Instructions:	Manually execute this test per the test steps.
Test Procedures:	1. For each API listed in the Security Target, perform a search for its public API documentation. 2. Validate that each API is supported.
Test Results:	This activity passes as the evaluator reviewed the public API documentation and found all API are supported.

4.3.6.4 FPT_IDV_EXT.1(Android), (iOS), & (iPadOS)

060	[APP_PP]FPT_IDV_EXT.1.1(Android), (iOS), & (iPadOS) – Software Identification and Versions
Test Purpose:	The evaluator shall install the application, then check for the existence of version information. If SWID tags is selected the evaluator shall check for a .swidtag file. The evaluator shall open the file and verify that is contains at least a SoftwareIdentity element and an Entity element.
Test Instructions:	Manually execute this test per the test steps.
Test Procedures:	NOTE: This was performed in conjunction with FPT_TUD_EXT.1.2. Android: 1. After the installation of the TOE application, check for the existence of version information. 2. Verify that the version number follows the “YY.MM.PP.BB” scheme and matches the expected running version. iOS / iPadOS: 1. After the installation of the TOE application, check for the existence of version information. 2. Verify that the version number follows the “YY.MM.PP” scheme and matches the expected running version.
Test Results:	This activity passes as the evaluator observed the TOE displaying the version numbers in the format defined for each platform.

4.3.6.5 FPT_LIB_EXT.1

061	[APP_PP]FPT_LIB_EXT.1.1 – Use of Third Party Libraries
Test Purpose:	The evaluator shall install the application and survey its installation directory for dynamic libraries. The evaluator shall verify that libraries found to be packaged with or employed by the application are limited to those in the assignment.
Test Instructions:	Manually execute this test per the test steps.
Test Procedures:	1. Start the mobsf process. 2. Navigate to the mobsf server. 3. Upload the target APK (for Android). 4. Wait for the analysis to complete.

	5. Examine the "DYNAMIC LIBRARY & FRAMEWORK BINARY ANALYSIS" section for the list of dynamic libraries found by scanner. 6. Extract list of dynamic libraries from scanner output 7. Compare list to the list in the ST to ensure all found libraries are declared in ST. 8. Repeat Steps 1 – 7 using IPA (for iOS/iPadOS) file for analysis.
Test Results:	This activity passes as the evaluator observed that the dynamic libraries found on the platforms are consistent with the declared library list in the ST.

4.3.6.6 FPT_TUD_EXT.1

062	**[APP_PP]FPT_TUD_EXT.1.1 – Integrity for Installation and Update**
Test Purpose:	The evaluator shall check for an update using procedures described in either the application documentation or the platform documentation and verify that the application does not issue an error. If it is updated or if it reports that no update is available this requirement is considered to be met.
Test Instructions:	Manually execute this test per the test steps.
Test Procedures:	1. Using the procedures described in the documentation, check for an update to the TOE application. 2. Verify that the TOE does not issue an error.
Test Results:	This activity passes as the evaluator observed that the TOE performs a version status check and declares to the end user if there is an update available.

063	**[APP_PP]FPT_TUD_EXT.1.2 – Integrity for Installation and Update**
Test Purpose:	The evaluator shall query the application for the current version of the software according to the operational user guidance. The evaluator shall then verify that the current version matches that of the documented and installed version.
Test Instructions:	Manually execute this test per the test steps.
Test Procedures:	1. Query the TOE application for its version: a. Authenticate to the TOE application. b. Tap "Settings" > "About". 2. Verify that it matches the installed version 3. Verify that it matches the documented version.
Test Results:	This activity passes as the evaluator observed that the TOE displayed the correct version that was installed and displayed the version in the correct format declared for the platform.

064	**[APP_PP]FPT_TUD_EXT.1.3 – Integrity for Installation and Update**
Test Purpose:	The evaluator shall verify that the application's executable files are not changed by the application. Platforms: Apple iOS... The evaluator shall consider the requirement met because the platform forces applications to write all data within the application working directory (sandbox). For all other platforms, the evaluator shall perform the following test: Test 1: The evaluator shall install the application and then locate all of its

	executable files. The evaluator shall then, for each file, save off either a hash of the file or a copy of the file itself. The evaluator shall then run the application and exercise all features of the application as described in the ST. The evaluator shall then compare each executable file with the either the saved hash or the saved copy of the files. The evaluator shall verify that these are identical.
Test Instructions:	Manually execute this test per the test steps.
Test Procedures:	Android: Install the TOE application on the device. Connect the mobile device to the test machine. Execute the following commands on the test machine to obtain the installation path of the TOE application on the mobile device: adb shell pm path com.boxer.email Execute the following commands on the test machine: adb shell run-as com.boxer.email cd /data/app/com.boxer.email-<uniquestring> find . -type f -print0 | xargs -0 sha256sum Record the hashes of the files. Launch the TOE application. Perform operations that stimulate the TOE application: Send a signed and encrypted e-mail. Receive a signed and encrypted e-mail. Send an unsigned and non-encrypted e-mail. Receive an unsigned and non-encrypted e-mail. Repeat Step 4. Verify that the hashes collected prior to running the TOE application are identical to the hashes collected after running and exercising the TOE application. iOS / iPadOS: Per the test assurance activity, “the evaluator shall consider the requirement met because the platform forces applications to write all data within the application working directory (sandbox).”
Test Results:	This activity passes as the evaluator observed that no executables were changed as a result of exercising the TOE.

4.3.6.7 FPT_TUD_EXT.2

065	**[APP_PP]FPT_TUD_EXT.2.1 – Integrity for Installation and Update – TD0628**

Test Purpose:	If a container image is claimed the evaluator shall verify that application updates are distributed as container images. If the format of the platform-supported package manager is claimed, the evaluator shall verify that application updates are distributed in the correct format. This varies per platform: Platforms: Android... The evaluator shall ensure that the application is packaged in the Android application package (APK) format. Platforms: Apple iOS... The evaluator shall ensure that the application is packaged in the IPA format.
Test Instructions:	Manually execute this test per the test steps.
Test Procedures:	Android: 1. Inspect the TOE application package APK file. 2. Verify that it has file signature “50 4b 03 04”. iOS / iPadOS: 1. Inspect the TOE application package IPA file. 2. Verify that it has file signature “50 4b 03 04”.
Test Results:	This activity passes as the evaluator observed, based on the file signatures, the Android package was found to be an APK and the iOS package was found to be an IPA

066	**[APP_PP]FPT_TUD_EXT.2.2 – Integrity for Installation and Update – TD0664**
Test Purpose:	Platforms: Android... The evaluator shall consider the requirement met because the platform forces applications to write all data within the application working directory (sandbox). Platforms: Apple iOS... The evaluator shall consider the requirement met because the platform forces applications to write all data within the application working directory (sandbox).
Test Instructions:	Manually execute this test per the test steps.
Test Procedures:	Android: N/A – Per test assurance activity, this requirement is met because the platform forces applications to write all data within the application working directory (sandbox). iOS / iPadOS: N/A – Per test assurance activity, this requirement is met because the platform forces applications to write all data within the application working directory (sandbox).
Test Results:	This activity is considered satisfied.

4.3.7 Trusted Path/Channel

4.3.7.1 FTP_DIT_EXT.1(Android), (iOS), & (iPadOS)

068	[APP_PP]FTP_DIT_EXT.1.1(Android), (iOS), & (iPadOS) – Protection of Data in Transit
Test Purpose:	The evaluator shall perform the following tests. Test 1: The evaluator shall exercise the application (attempting to transmit data; for example by connecting to remote systems or websites) while capturing packets from the application. The evaluator shall verify from the packet capture that the traffic is encrypted with HTTPS, TLS, DTLS, SSH, or IPsec in accordance with the selection in the ST.
Test Instructions:	Manually execute this test per the test steps.
Test Procedures:	1. Begin capturing packets between the TOE and the remote TLS server. 2. Stimulate TOE to force a connection to the remote TLS server. 3. Stop capturing packets between the TOE and the remote TLS server. 4. Analyze packet capture to verify that the connection was successful and that it was encrypted.
Test Results:	This activity passes as the evaluator observed that the communication channels were protected using TLS.

069	[APP_PP]FTP_DIT_EXT.1.1(Android), (iOS), & (iPadOS) – Protection of Data in Transit
Test Purpose:	The evaluator shall perform the following tests. Test 2: The evaluator shall exercise the application (attempting to transmit data; for example by connecting to remote systems or websites) while capturing packets from the application. The evaluator shall review the packet capture and verify that no sensitive data is transmitted in the clear.
Test Instructions:	Manually execute this test per the test steps.
Test Procedures:	1. Begin capturing packets between the TOE and the remote TLS server. 2. Stimulate TOE to force a connection to the remote TLS server. 3. Stop capturing packets between the TOE and the remote TLS server. 4. Analyze packet capture to verify that the connection was successful, encrypted, and did not transmit any sensitive data in the clear.
Test Results:	This activity passes as the evaluator observed that the only plaintext data transferred were OCSP requests/response data, which is not sensitive data, and is expected to be communicated plaintext via HTTP per the RFC for OCSP.

070	[APP_PP]FTP_DIT_EXT.1.1(Android), (iOS), & (iPadOS) – Protection of Data in Transit
Test Purpose:	The evaluator shall perform the following tests. Test 3: The evaluator shall inspect the TSS to determine if user credentials are transmitted. If credentials are transmitted the evaluator shall set the credential to a known value. The evaluator shall capture packets from the application while causing credentials to be transmitted as described in the TSS. The evaluator shall perform a string search of the captured network packets and verify that the plaintext credential previously set by the evaluator is not found.
Test Instructions:	Manually execute this test per the test steps.
Test Procedures:	1. Begin capturing packets between the TOE and the remote TLS server. 2. Stimulate TOE to force a connection to the remote TLS server. 3. Stop capturing packets between the TOE and the remote TLS server. 4. Analyze packet capture to verify that the known plaintext value for the TOE application, “qwer1”, and the Exchange server mailbox password,

	"P@ssw0rd", are not found in the packet capture.
Test Results:	This activity passes as the evaluator observed passwords were not sent in plaintext. A search of a predefined password within the captured traffic returned with no results.

071	**[APP_PP]FTP_DIT_EXT.1.1(Android), (iOS), & (iPadOS) – Protection of Data in Transit**
Test Purpose:	The evaluator shall perform the following tests. Platforms: Android... If "not transmit any data" is selected, the evaluator shall ensure that the application's AndroidManifest.xml file does not contain a uses-permission or uses-permission-sdk-23 tag containing android:name="android.permission.INTERNET". In this case, it is not necessary to perform the above Tests 1, 2, or 3, as the platform will not allow the application to perform any network communication. Platforms: Apple iOS... If "encrypt all transmitted data" is selected, the evaluator shall ensure that the application's Info.plist file does not contain the NSAllowsArbitraryLoads or NSExceptionAllowsInsecureHTTPLoads keys, as these keys disable iOS's Application Transport Security feature.
Test Instructions:	Manually execute this test per the test steps.
Test Procedures:	Android: N/A – This portion of the test assurance activity does not apply because "not transmit any data" was not selected. iOS / iPadOS: N/A – This portion of the test assurance activity does not apply because "encrypt all transmitted data" was not selected.
Test Results:	This activity is considered satisfied.

4.3.7.2 FTP_ITC_EXT.1

072	**[EC_EP]FTP_ITC_EXT.1.1 – Inter-TSF Trusted Channel – TD0414**
Test Purpose:	The evaluator shall also perform the following tests: Test 1: The evaluators shall ensure that the email client is able to initiate or receive communications using any selected or assigned protocols specified in the requirement over TLS, setting up the connections as described in the operational guidance and ensuring that communication is successful.
Test Instructions:	Manually execute this test per the test steps.
Test Procedures:	1. Begin capturing packets between the TOE and the remote Exchange server. 2. Stimulate TOE client to force a connection to the remote Exchange server. 3. Stop capturing packets between the TOE and the remote Exchange server. 4. Verify that the TOE successfully established a connection with the remote

	Exchange server, that the communications are not sent in plaintext, and that the protocol in use is TLS.
Test Results:	This activity passes as the evaluator observed that the TOE could initiate and receive communication using TLS, which is the only declared protocol.

073	**[EC_EP]FTP_ITC_EXT.1.2 – Inter-TSF Trusted Channel – TD0414**
Test Purpose:	The evaluator shall also perform the following tests: Test 2: The evaluators shall ensure that the email client is able to initiate or receive communications with a Mail Transfer Agent using any assigned protocols specified in the requirement over TLS, setting up the connections as described in the operational guidance and ensuring that communication is successful.
Test Instructions:	Manually execute this test per the test steps.
Test Procedures:	1. Begin capturing packets between the TOE and the remote Mail Transfer Agent. 2. Stimulate TOE client to force a connection to the remote Mail Transfer Agent. 3. Stop capturing packets between the TOE and the remote Mail Transfer Agent. 4. Verify that the TOE successfully established a connection with the remote Mail Transfer Agent, that the communications are not sent in plaintext, and that the protocol in use is TLS.
Test Results:	This activity passes as the evaluator observed that the communication channel between the TOE and the Mail Transfer Agent was protected using TLS, which is the only declared protocol.

074	**[EC_EP]FTP_ITC_EXT.1.2 – Inter-TSF Trusted Channel**
Test Purpose:	The evaluator shall also perform the following tests: Test 3: The evaluator shall ensure, for each communication channel with an authorized IT entity in tests 1 and 2, the channel data is not sent in plaintext. To perform this test, the evaluator shall use a sniffer and a packet analyzer. The packet analyzer must indicate that the protocol in use is TLS.
Test Instructions:	Manually execute this test per the test steps.
Test Procedures:	NOTE: This test is met by testing performed in [EC_EP]FTP_ITC_EXT.1 – Test Case 072 and [EC_EP]FTP_ITC_EXT.1 – Test Case 073.
Test Results:	This activity is considered satisfied.

5 Evaluation Activities for SARs

This section addresses assurance activities that are defined in the *Protection Profile for Application Software Version 1.4* [AppPP] that correspond with Security Assurance Requirements. There are no additional SARs defined by the Application Software Extended Package for Email Clients.

AGD_OPE.1 – *"Some of the contents of the operational guidance will be verified by the evaluation activities in 5.1 Security Functional Requirements and evaluation of the TOE according to the [CEM]. The following additional information is also required.*

If cryptographic functions are provided by the TOE, the operational guidance shall contain instructions for configuring the cryptographic engine associated with the evaluated configuration of the TOE. It shall

provide a warning to the administrator 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 by verifying a digital signature – this may be done by the TOE or the underlying platform.

The evaluator shall verify that this process includes the following steps:

- *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 digital signature. The TOE will likely contain security functionality that does not fall in the scope of evaluation under this PP. The operational guidance shall make it clear to an administrator which security functionality is covered by the evaluation activities."*

Section 6.2 of the AGD contains instructions for configuring the cryptographic engine associated with the evaluated configuration of the TOE. Specifically, Section 6.2.2 covers the process for configuring the S/MIME encryption, message digest, and signature algorithms for S/MIME.

Section 7 of the AGD provides a warning that the information contained within the AGD is to "discuss only actions that are required as part of the 'evaluated configuration'." Section 6.2 provides a note stating that "the use of other cryptographic engines and cryptographic settings were not evaluated nor tested during the Common Criteria evaluation of the TOE."

Section 7.3 of the AGD describes the process for verifying updates to the TOE by verifying a digital signature where the update is initially signed by the developer and then once uploaded to the platform application store, the store verifies that signature and then signs the update with its own digital signature. The AGD advises users to verify the version to ensure that the update was successfully installed.

Updates to the TOE are provided by the Google Play Store (Android) or Apple Store (iOS) over HTTPS/TLS. The user is able to check for updates to the application by navigating to the platform provided Google Play Store (Android) or the Apple Store (iOS). When the update is sent to the mobile device, the mobile device will verify the signature from the platform application store.

Section 2 of the AGD states that any functionality that is not described here or in the VMware Workspace ONE Boxer Security Target was not evaluated and should be exercised at the user's risk.

AGD_PRE.1 – *"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 claimed for the TOE in the ST."*

Section 5.1 of the AGD describes the TOE components in the evaluated configuration: VMware Workspace ONE Boxer Email Client Version 23.11 Application for iOS/iPadOS 16 and VMware Workspace ONE Boxer Email Client Version 23.11 Application for Android 13.0. Section 5.3 of the AGD contains instructions for the Security Administrator to ensure that the operational environment will fulfill its role in supporting the TOE. These instructions match the assumptions for the TOE's operational environment in Section 4.3 of the ST.

ALC_CMC.1 – *"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."

The evaluation team verified that the Security Target (ST), TOE, and Supplemental Administrative Guidance (AGD) were labeled consistently to correctly identify the operational environment and TOE hardware and software versions in the CC evaluation.

Specifically, Section 1.2 of the ST states in the TOE Reference that the TOE is the "VMware Workspace ONE Boxer Email Client Version 23.11". Section 2 of the AGD states that the AGD is intended for VMware Workspace ONE Unified Endpoint Management (UEM) administrators and users responsible for deploying, configuring, and/or operating VMware Workspace ONE Boxer Email Client Version 23.11. Finally, the product web site, https://www.vmware.com/products/workspace-one/enterprise-email.html, contains identifying product information including a whitepaper, infographic, and description of Enterprise Email: Workspace ONE Boxer". The ST states in the TOE Overview that "Boxer is an email client application software product that is installed on a mobile device platform. The Boxer application containerizes enterprise data from personal data that resides on the user's mobile device. Boxer supports the use of Microsoft Exchange (using ActiveSync and/or Exchange Web Services), Office 365, Outlook, Gmail, Yahoo, G Suite and Lotus Notes, and Cloud email services."

All of this information as stated above provides sufficient context to accurately identify the TOE as such in the ST, AGD, and vendor web site.

ALC_CMS.1 – *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 evaluation activity for ALC_CMC.1), the evaluator implicitly confirms the information required by this component. Life-cycle support is targeted aspects of the developer's life-cycle and instructions to providers of applications for the developer's devices, rather than an in-depth examination of the TSF manufacturer'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.*

The evaluator shall ensure that the developer has identified (in guidance documentation for application developers concerning the targeted platform) one or more development environments appropriate for use in developing applications for the developer's platform. For each of these development environments, the developer shall provide information on how to configure the environment to ensure that buffer overflow protection mechanisms in the environment(s) are invoked (e.g., compiler flags). The evaluator shall ensure that this documentation also includes an indication of whether such protections are on by default, or have to be specifically enabled. The evaluator shall ensure that the TSF is uniquely identified (with respect to other products from the TSF vendor), and that documentation provided by the developer in association with the requirements in the ST is associated with the TSF using this unique identification."

Section 7.2.4 of the AGD states "the TOE does not support the installation of trusted or untrusted add-ons." As such, Boxer is an end-user application that does not support add-ons and is not a framework for other developers to be able to add or create functions within the application. Therefore, there are no publicly available development documents. The Boxer application is uniquely identified and under configuration management control by VMware.

ALC_TSU_EXT.1 – *"The evaluator shall verify that the TSS contains a description of the timely security update process used by the developer to create and deploy security updates. The evaluator shall verify that this description addresses the entire application. The evaluator shall also verify that, in addition to the TOE developer's process, any third-party processes are also addressed in the description. The evaluator shall also verify that each mechanism for deployment of security updates is described.*

The evaluator shall verify that, for each deployment mechanism described for the update process, the TSS lists a time between public disclosure of a vulnerability and public availability of the security update to the TOE patching this vulnerability, to include any third-party or carrier delays in deployment. The evaluator shall verify that this time is expressed in a number or range of days.

The evaluator shall verify that this description includes the publicly available mechanisms (including either an email address or website) for reporting security issues related to the TOE. The evaluator shall verify that the description of this mechanism includes a method for protecting the report either using a public key for encrypting email or a trusted channel for a website."

The evaluation team verified that the Security Target (ST) contains in the TSS section for FPT_TUD_EXT.1 (Section 8.6.6.1) a description of the timely security update process used by the developer to create and deploy security updates:

> As part of providing timely security updates, VMware provides customers with a support section on VMware.com where they have the ability to submit support issues. This is an HTTPS site that requires user authentication prior to use. Any feedback that necessitates a fix will result in an update to Boxer itself so there is no third-party update process to consider when updating the TOE. High severity issues can result in a patch release as soon as remediation is available. Lower severity issues will be incorporated into the next monthly release. Security fixes will be released as new packages in the same manner as any feature updates (see discussion on FPT_TUD_EXT.1 above). The TOE contains a number of components, including third-party components that VMware does not have control over the implementation of. Any implementation flaws are expected to be addressed within 90 days of reporting. Customers are notified of security-related fixes on the VMware customer portal.

This adequately addresses the entire application, including that there are no third-party update processes to consider when updating the TOE. It also describes the process by which security updates are retrieved, a specific timeframe (in days) for which reported flaws are expected to be addressed, and that the VMware website reporting mechanism uses a trusted HTTPS channel requiring user authentication to submit issues.

AVA_VAN.1 – *"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 find vulnerabilities that have been found in similar applications with a particular focus on network protocols the application uses and document formats it parses. The evaluator shall also run a virus scanner with the most current virus definitions against the application files and verify that no files are flagged as malicious. 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. 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."

***"For Windows, Linux, macOS and Solaris:** The evaluator shall also run a virus scanner with the most current virus definitions against the application files and verify that no files are flagged as malicious."*

The evaluation team reviewed vendor documentation, formulated hypotheses, performed vulnerability analysis, and documented the hypotheses and analysis in accordance with the [AppPP] requirements. Keywords were identified based upon review of the Security Target and AGD. The following keywords were identified:

Keyword	Description
Boxer	This is a generic term for searching for known vulnerabilities for the specific product.

Keyword	Description
ActiveSync	This is a generic term for searching for known vulnerabilities for the network protocol used by the TOE.
OpenSSL Android: (version 1.0.2zi)	This is a generic term for searching for known vulnerabilities for the cryptographic library used by the TOE application.
WebView	This is a generic term for searching for known vulnerabilities for the email document (rich text/HTML) viewer used by the TOE application (Android).
Polaris Office (version 8.1)	This is a generic term for searching for known vulnerabilities for the email attachment viewer used by the TOE application (Android).
WKWebView	This is a generic term for searching for known vulnerabilities for the email document (HTML) and attachment viewer used by the TOE application (Android, iOS, iPadOS).
Additional third party libraries listed in ST.	Separate tables for the Android and iOS/iPadOS applications listing third party libraries.

These keywords were used individually and as part of various permutations and combinations to search for vulnerabilities on public vulnerability sources (updated April 20, 2024). The following public vulnerability sources were searched:

a) NIST National Vulnerabilities Database (can be used to access CVE and US-CERT databases identified below): https://web.nvd.nist.gov/view/vuln/search

b) Common Vulnerabilities and Exposures: http://cve.mitre.org/cve/
https://www.cvedetails.com/vulnerability-search.php

c) US-CERT: http://www.kb.cert.org/vuls/html/search

e) Tenable Network Security http://nessus.org/plugins/index.php?view=search

f) Tipping Point Zero Day Initiative http://www.zerodayinitiative.com/advisories

g) Offensive Security Exploit Database: https://www.exploit-db.com/

h) Rapid7 Vulnerability Database: https://www.rapid7.com/db/vulnerabilities

Upon the completion of the vulnerability analysis research, the team had identified several generic vulnerabilities upon which to build a test suite. These tests were created specifically with the intent of exploiting these vulnerabilities within the TOE or its configuration. Testing that was conducted under the functional testing that would have been duplication of a vulnerability tests were not re-run. This left one remaining exploit to further explore: malicious binary.

The team tested the following areas:

- Malicious Binary Analysis
  This test analyzes the TOE binary using the most current OSINT threat intelligence data against the TOE Android and iOS package files and verify that they do not contain references to network addresses that are flagged as malicious according to the threat intelligence database.

The evaluation team determined that no residual vulnerabilities exist that are exploitable by attackers with Basic Attack Potential.

6 Conclusions

The TOE was evaluated against the ST and has been found by this evaluation team to be conformant with the ST. The overall verdict for this evaluation is: Pass.

7 Glossary of Terms

Acronym	Definition
CC	Common Criteria
CLI	Command-Line Interface
cPP	collaborative Protection Profile
FTP	File Transfer Protocol
GUI	Graphical User Interface
HTTPS	Hypertext Transfer Protocol Secure
NIAP	National Information Assurance Partnership
NTP	Network Time Protocol
OS	Operating System
PP	Protection Profile
RBG	Random Bit Generator
SFR	Security Functional Requirement
SHA	Secure Hash Algorithm
SHS	Secure Hash Standard
SSH	Secure Shell
ST	Security Target
TLS	Transport Layer Security
TOE	Target of Evaluation
TSF	TOE Security Function
UI	User Interface

Table 7-1: Acronyms

Term	Definition
Administrator	A user who is assigned the Admin role on the TOE and has the ability to manage the TSF.
Security Administrator	The claimed Protection Profile defines a single Security Administrator role that is authorized to manage the TOE and its data. This TOE defines three separate user roles, but only the most privileged role (Admin) is authorized to manage the TOE's security functionality and is therefore considered to be the Security Administrator for the TOE.
Trusted Channel	An encrypted connection between the TOE and a system in the Operational Environment.
Trusted Path	An encrypted connection between the TOE and the application a Security Administrator uses to manage it (web browser, terminal client, etc.).
User	In a CC context, any individual who has the ability to access the TOE functions or data.

Table 7-2: Terminology