VMware Workspace ONE Boxer Email Client 5.4

Security Target

ST Version: 1.0
June 13, 2019

VMware
1155 Perimeter Center West
Suite 100
Atlanta, GA 30338

Prepared By:

Booz | Allen | Hamilton
delivering results that endure

Cyber Assurance Testing Laboratory
1100 West St.
Laurel, MD 20707

Table of Contents

Table of Figures

Table of Tables

1 Security Target Introduction

This chapter presents the Security Target (ST) identification information and an overview. An ST contains the Information Technology (IT) security requirements of an identified Target of Evaluation (TOE) and specifies the functional and assurance security measures offered by the TOE.

1.1 ST Reference

This section provides information needed to identify and control this ST and its Target of Evaluation.

1.1.1 ST Identification

ST Title:	VMware Workspace ONE Boxer Email Client 5.4 Security Target
ST Version:	1.0
ST Publication Date:	June 13, 2019
ST Author:	Booz Allen Hamilton

1.1.2 Document Organization

Chapter 1 of this document provides identifying information for the ST and TOE as well as a brief description of the TOE and its associated TOE type.

Chapter 2 describes the TOE in terms of its physical boundary, logical boundary, exclusions, and dependent Operational Environment components.

Chapter 3 describes the conformance claims made by this ST.

Chapter 4 describes the threats, assumptions, objectives, and organizational security policies that apply to the TOE.

Chapter 5 defines extended Security Functional Requirements (SFRs) and Security Assurance Requirements (SARs).

Chapter 6 describes the SFRs that are to be implemented by the TSF.

Chapter 7 describes the SARs that will be used to evaluate the TOE.

Chapter 8 provides the TOE Summary Specification, which describes how the SFRs that are defined for the TOE are implemented by the TSF.

1.1.3 Terminology

This section defines the terminology used throughout this ST. The terminology used throughout this ST is defined in Table 1 and 2. These tables are to be used by the reader as a quick reference guide for terminology definitions.

Term	Definition
Administrator	An individual that has the ability to manage some aspect of mobile device configuration using the VMware Workspace ONE Unified Endpoint Management (UEM) console. UEM is a Mobile Device Management (MDM) product that contains a server and an agent that resides on the mobile device.
Application Management	The class of TOE Administrators that provides the ability to deploy and manage internal and public apps for managed devices.
End User	An individual who possesses a mobile device that is managed by VMware Workspace ONE Unified Endpoint Management (UEM).

Table 1: Customer Specific Terminology

Term	Definition
Authorized Administrator	The claimed Protection Profile defines an Authorized Administrator role that is authorized to manage the TOE and its data.
Security Administrator	Synonymous with Authorized Administrator.
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 an Authorized Administrator uses to manage it (web browser, terminal client, etc.).
User	In a CC context, any individual who has the ability to manage TOE functions or data.

Table 2: CC Specific Terminology

1.1.4 Acronyms

The acronyms used throughout this ST are defined in Table 3. This table is to be used by the reader as a quick reference guide for acronym definitions.

Acronym	Definition
CA	Certificate Authority
CC	Common Criteria
CPU	Central Processing Unit
GUI	Graphical User Interface
HTTP	Hypertext Transfer Protocol
HTTPS	Hypertext Transfer Protocol Secure over a bidirectional TLS encrypted tunnel
IP	Internet Protocol
IT	Information Technology
LDAP	Lightweight Directory Access Protocol
MAS	Mobile Application Store
MDM	Mobile Device Management
NIAP	National Information Assurance Partnership
OCSP	Online Certificate Status Protocol
OS	Operating System
PP	Protection Profile

SAR	Security Assurance Requirement
SFP	Security Function Policy
SFR	Security Functional Requirement
SSL	Secure Sockets Layer
ST	Security Target
TCP	Transmission Control Protocol
TLS	Transport Layer Security
TOE	Target of Evaluation
TSF	TOE Security Function
UEM	Unified Endpoint Management

Table 3: Acronym Definition

1.1.5 Reference

[1] Protection Profile for Application Software, version 1.2 [APP_PP]

[2] Application Software Extended Package for Email Clients version 2.0 [EC_EP]

[3] Common Criteria for Information Technology Security Evaluation – Part 1: Introduction and general model, dated April 2017, version 3.1, Revision 5, CCMB-2012-009-001

[4] Common Criteria for Information Technology Security Evaluation – Part 2: Security functional components, dated April 2017, version 3.1, Revision 5, CCMB-2012-009-002

[5] Common Criteria for Information Technology Security Evaluation – Part 3: Security assurance components, dated April 2017, version 3.1, Revision 5, CCMB-2012-009-003

[6] Common Methodology for Information Technology Security Evaluation – Evaluation Methodology, dated April 2017, version 3.1, Revision 5, CCMB-2012-009-004 [CEM]

[7] NIST Special Publication 800-56B Recommendation for Pair-Wise Key Establishment Schemes Using Integer Factorization Cryptography, August 2009

[8] NIST Special Publication 800-38A Recommendation for Block Cipher Modes of Operation, December 2001

[9] FIPS PUB 140-2 Federal Information Processing Standards Publication Security Requirements for Cryptographic Modules May 25, 2001

[10] FIPS PUB 180-3 Federal Information Processing Standards Publication Secure Hash Standard (SHS) October 2008

[11] FIPS PUB 180-4 Federal Information Processing Standards Publication Secure Hash Standard (SHS) March 2012

[12] FIPS PUB 186-4 Federal Information Processing Standards Publication Digital Signature Standard July 2013

[13] FIPS PUB 197 Advanced Encryption Standard November 26, 2001

[14] FIPS PUB 198-1 Federal Information Processing Standards Publication The Keyed-Hash Message Authentication Code (HMAC) July 2008

[15] Samsung Electronics Co., Ltd. Samsung Galaxy Devices on Android 8.0 (MDFPP30/WLANCEP10) Security Target (VID10898)

[16] Apple iOS 11 PP_MD_V3.0, EP_MDM_AGENT_V3.0, & PP_WLAN_CLI_EP Security Target (VID10851)

[17] VMware Workspace ONE Boxer Admin Guide – 1/15/2019

[18] VMware Workspace ONE Boxer for Android User Guide – 2/10/2019

[19] VMware Workspace ONE Boxer for iOS User Guide – 3/18/2019

1.2 TOE Reference

The TOE is the VMware Workspace ONE Boxer Email Client version 5.4

1.3 TOE Overview

The TOE is the VMware Workspace ONE Boxer Email Client product referred to as Boxer or TOE from this point forward. 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 Exchange, Office 365, Outlook, Gmail, Yahoo and Cloud email services. Enterprise management support only applies to the use of Exchange.

In the evaluated configuration, the TOE is installed on a mobile device running iOS 11 (VID10851) as well as a mobile device host running Android 8.0 (VID10898). The mobile device that the TOE is installed on is managed by a Mobile Device Management software product called VMware Workspace ONE Unified Endpoint Management (UEM). UEM consists of a server and an agent that resides on the mobile device. The UEM agent is used to enroll the mobile device with the UEM server so that it can be managed by the UEM server. Also, the UEM agent consumes policy and configuration information for the device and VMware applications, such as Boxer, operating on the device, as well as providing status and policy information about the mobile device to the UEM server. The operating system, UEM agent, and UEM server are considered part of the operational environment.

Boxer uses ActiveSync to communicate with the Exchange server and is protected using TLS v1.2. The Exchange server resides in the operational environment and is for sending and receiving enterprise data such as email, calendar information and appointment data. Whether installed on an Android or iOS device, the application validates the certificates using OCSP. The OCSP responder is also considered part of the operational environment.

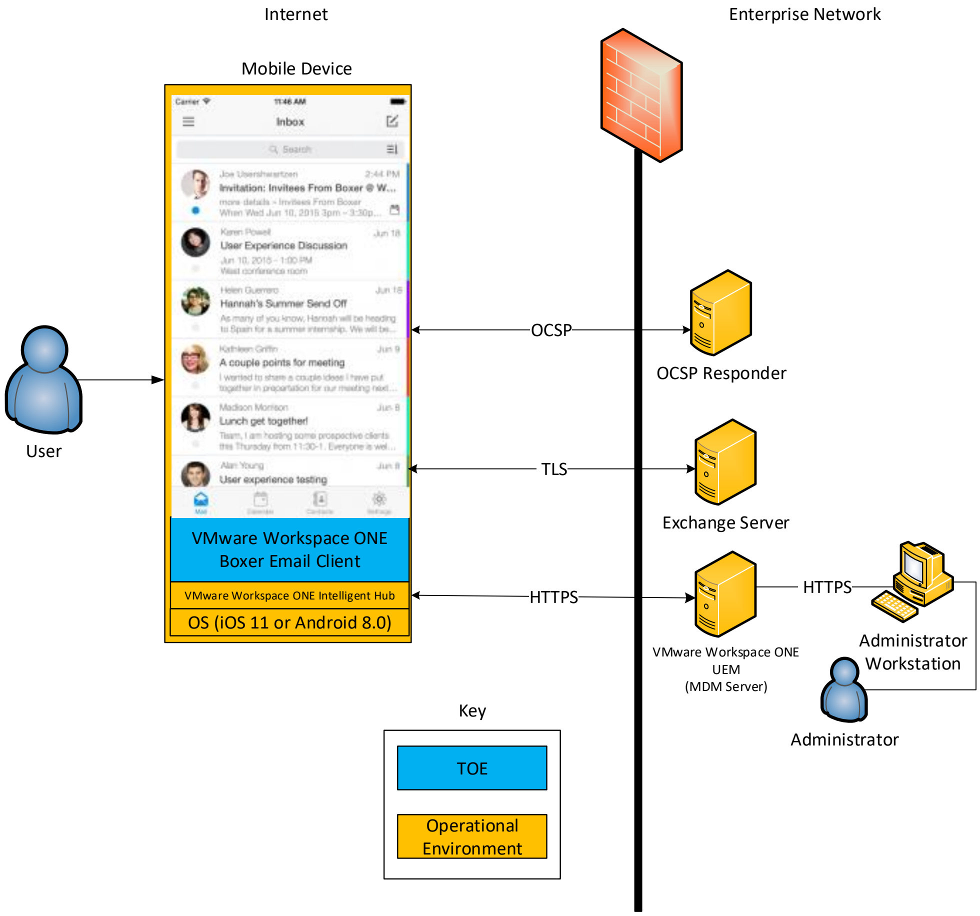
Internet
Enterprise Network
Mobile Device
Inbox
Search
Invitation: Invitees From Boxer @ W...
User Experience Discussion
Hannah's Summer Send Off
A couple points for meeting
Lunch get together!
User experience testing
VMware Workspace ONE Boxer Email Client
VMware Workspace ONE Intelligent Hub
OS (iOS 11 or Android 8.0)
User
OCSP
OCSP Responder
TLS
Exchange Server
HTTPS
VMware Workspace ONE UEM (MDM Server)
HTTPS
Administrator Workstation
Administrator
Key
TOE
Operational Environment

Figure 1: TOE Boundary

As depicted in Figure 1, the TOE resides on the mobile device host running iOS 11 or Android 8.0. The mobile devices are required to be under the control of the Workspace ONE UEM product. The mobile devices, running the UEM agent, will communicate with the UEM server to consume policy and configuration information for the device and VMware applications, such as the Boxer email client.

Boxer email client communicates with the Exchange server over TLS v1.2 using mutual authentication with X.509v3 certificates for authentication. There is also a communication channel between the mobile device platform and the OCSP responder to check certificate revocation status.

The TOE is the VMware Workspace ONE Boxer Email Client. The TOE interfaces used to communicate with the Exchange server and OCSP responder are subject to evaluation testing. The mobile device, OS, UEM agent, UEM server, UEM Administrator Workstation (aka UEM Console), Exchange server, and OCSP responder are operational environment components.

1.4 TOE Type

The TOE is an application software email client product that is installed on mobile devices. The [APP_PP] states the following:

“The application, which consists of the software provided by its vendor, is installed onto the filesystem provided by the operating system. It executes on the platform, which may be an operating system, an execution environment, or some combination of these.

Applications include a diverse range of software such as office suites, thin clients, PDF readers, and downloadable smartphone apps. The TOE includes any software in the application installation package, even those pieces that may extend the functionality of the underlying platform, such as kernel drivers.”

The [EC_EP] states the following:

“Email clients are user applications that provide functionality to send, receive, access and manage email. The complexity of email content and email clients has grown over time. Modern email clients can render HTML as well as plaintext, and may include functionality to display common attachment formats, such as Adobe PDF and Microsoft Word documents. Some email clients allow their functionality to be modified by users through the addition of add-ons. Protocols have also been defined for communicating between email clients and servers. Some clients support multiple protocols for doing the same task, allowing them to be configured according to email server specifications.”

The Application Software Email Clients TOE type is justified because the TOE is an email client application that allows the user to receive, send, manage, and access enterprise email on their mobile device.

2 TOE Description

This section provides a description of the TOE in its evaluated configuration. This includes the physical and logical boundaries of the TOE.

2.1 Evaluated Components of the TOE

The following table describes the TOE components in the evaluated configuration:

Component	Definition
VMware Workspace ONE Boxer Email Client v5.4 Application on Apple iOS 11*	VMware Email Client Application
VMware Workspace ONE Boxer Email Client v5.4 Application on Android 8.0*	VMware Email Client Application

Table 4: Evaluated Components of the TOE

*VID10851 certified iOS 11 and VID10898 certified Android 8.0.

As shown in Figure 1, the TOE boundary on the end user mobile devices includes only the VMware Workspace ONE Boxer Email Client application (Boxer from this point forward). The mobile device and the OS that the application is installed on are considered a part of the operational environment. The devices and the OS have been evaluated against the Mobile Device Fundamentals Protection Profile under the Validation ID numbers identified in Table 4 above.

2.2 Components and Applications in the Operational Environment

The following table lists components and applications in the environment that the TOE relies upon in order to function properly:

Component	Definition
OCSP Responder	A server deployed within the Operational Environment which confirms the validity and revocation status of certificates.
VMware Workspace ONE UEM v9.4.0.0	The mobile device has the VMware Workspace ONE Intelligent Hub (UEM agent) installed and is managed by the VMware Workspace ONE UEM server (UEM server).
Windows Server 2012 R2 Exchange server	Exchange server for sending and receiving emails to and from the Operational Environment.
Mobile Device	The hardware that runs the OS in which the application is installed on.

Table 5: Evaluated Components of the Operational Environment

2.3 Excluded from the TOE

The following optional products, components, and/or applications can be integrated with the TOE but are not included in the evaluated configuration. They provide no added security related functionality for the evaluated product. They are separated into three categories: not installed, installed but requires a separate license, and installed but not part of the TSF.

2.3.1 Not Installed

There are no components that are not installed.

2.3.2 Installed but Requires a Separate License

There are no excluded components that are installed and require a separate license.

2.3.3 Installed but Not Part of the TSF

This section contains functionality or components that are part of the purchased product but are not part of the TSF relevant functionality that is being evaluated as the TOE:

There are no discrete individual components that are excluded from the TSF. Note however that the logical boundary of the TOE only includes the functionality that are satisfied by the Security Functional Requirements in the claimed Protection Profiles. If the product provides functionality that is not used to satisfy any of these requirements, it is considered to be security-non-interfering functionality.

2.4 Physical Boundary

2.4.1 Hardware

The TOE is an application software product. All hardware that is present is part of the TOE's Operational Environment. In the evaluated configuration, the TOE is installed on a VID10851 certified iOS 11 device and VID10898 certified Android 8.0 device. For testing, this evaluation used a Samsung Galaxy S8+ (Android) and on an iPhone 8 (Apple).

2.4.2 Software

The Boxer application runs on a mobile device running Apple iOS 11 OS as well as a mobile device running Android 8.0. The mobile devices will also have the VMware Workspace ONE UEM agent installed, as the devices are managed by the VMware Workspace ONE UEM server.

2.5 Logical Boundary

The TOE is comprised of several security features. Each of these security features consists of several security functionalities, as identified below. This ST includes the security functional requirements from the Application Software Protection Profile v1.2 and the Email Client Extended Package. The security requirements that are derived from the Application Software Protection Profile are denoted with [APP_PP] and the requirements that are derived from the Application Software Extended Package for Email Clients are denoted with [EC_EP].

1. Cryptographic Support
2. User Data Protection
3. Identification and Authentication
4. Security Management
5. Privacy
6. Protection of the TSF
7. Trusted Path/Channels

2.5.1 Cryptographic Support

Depending on which OS the application is installed on, the TOE either invokes the underlying platform or implements its own cryptographic module to perform cryptographic services. All cryptographic mechanisms, whether platform or application provided, use DRBG functionality to support cryptographic operations. Cryptographic functionality includes encryption/decryption services, credential/key storage, key establishment, key destruction, hashing services, signature services, key-hashed message authentication, and key chaining using a password-based derivation function.

Cryptographic services for the application's S/MIME functionality and TLS communications are provided by the underlying platform when the application is installed on a device running the iOS. When installed on a device running the Android OS, the TOE invokes the underlying platform cryptographic libraries for TLS communications and implements an OpenSSL cryptographic module to perform the cryptographic functionality required to support S/MIME (consolidated Certificate number C631).

2.5.2 User Data Protection

The TOE uses S/MIME to digitally sign, verify, decrypt, and encrypt email messages. The TOE stores all application data in an encrypted Boxer database which is created on the mobile device during installation. The TOE requires that the host platform have full disk encryption enabled to securely store the data. The TOE restricts its network access and provides user awareness when it attempts to access hardware resources and sensitive data stored on the host platform. The TOE displays notification icons that show S/MIME status. Each status is shown as a different color so that the user can quickly identify any issues.

2.5.3 Identification and Authentication

The TOE validates the X.509v3 certificates for TLS communication to the Exchange server. X.509v3 certificates are also used for signing and encrypting emails for S/MIME. The TOE application, regardless of platform, performs the certificate validation using OCSP.

2.5.4 Security Management

The TOE enforces the application's enterprise policy set by the UEM administrator pushed out to the managed devices. The TOE does not use default passwords, and automatically installs and configures the application to protect itself and its data from unauthorized access while also implementing the recommended platform security mechanisms. Changing one's own password from the application is the only management function that can be performed by the owner/user of the mobile device with the TOE installed.

2.5.5 Privacy

The TOE does not transmit any personally identifiable information (PII) over the network unless voluntarily sent via free text email.

2.5.6 Protection of the TSF

The TOE does not support the installation of trusted or untrusted add-ons. The user is able to navigate the platform to check the version of the TOE and also check for updates to the application. All updates come from the Google Play Store (Android) or Apple Store (iOS). The digital signature of the updates is verified by the mobile device platform prior to being installed. The TOE does not replace or modify its own binaries without user interaction. The TOE implements anti-exploitation features, such as stack-based overflow protection, is compatible with security features provided by the OS, and will only use documented APIs and libraries.

2.5.7 Trusted Path/Channels

The TOE invokes the platform to provide the trusted communication channel between the TOE and the Exchange server. Communications is protected with TLS v1.2. Communication to the Exchange server uses ActiveSync to send and receive emails. The TOE, in conjunction with the platform, supports mutual authentication using X.509v3 certificates for TLS communications.

3 Conformance Claims

3.1 CC Version

This ST is compliant with Common Criteria for Information Technology Security Evaluation, Version 3.1 Revision 5 April 2017

3.2 CC Part 2 Conformance Claims

This ST and Target of Evaluation (TOE) is Part 2 extended to include all applicable NIAP and International interpretations through June 13, 2019.

3.3 CC Part 3 Conformance Claims

This ST and Target of Evaluation (TOE) are conformant to Part 3 extended to include all applicable NIAP and International interpretations through June 13, 2019.

3.4 PP Claims

This ST claims exact conformance to the following Protection Profiles:

- Protection Profile for Application Software Version 1.2 [APP_PP]
- Application Software Extended Package for Email Clients v2.0 [EC_EP]

3.5 Package Claims

The TOE claims exact compliance to the *Protection Profile for Application Software* and *The Application Software Extended Package for Email Clients*.

The TOE claims following optional SFRs that are defined in the appendices of the claimed PP:

[APP_PP] FCS_TLSC_EXT.2.1

[EC_EP] FCS_CKM_EXT.5

This does not violate the notion of exact conformance because the PP specifically indicates these as allowable options and provides both the ST author and evaluation laboratory with instructions on how these claims are to be documented and evaluated.

3.6 Package Name Conformant or Package Name Augmented

This ST and TOE claim exact conformance to the [APP_PP] and [EC_EP].

3.7 Conformance Claim Rationale

The [APP_PP] states the following:

“The application, which consists of the software provided by its vendor, is installed onto the filesystem provided by the operating system. It executes on the platform, which may be an operating system, an execution environment, or some combination of these.”

"Applications include a diverse range of software such as office suites, thin clients, PDF readers, and downloadable smartphone apps. The TOE includes any software in the application installation package, even those pieces that may extend the functionality of the underlying platform, such as kernel drivers."

The Application Software TOE type is justified because the TOE is software only and is able to be installed on a mobile device.

The [EC_EP] states the following:

"Email clients are user applications that provide functionality to send, receive, access and manage email. The complexity of email content and email clients has grown over time. Modern email clients can render HTML as well as plaintext, and may include functionality to display common attachment formats, such as Adobe PDF and Microsoft Word documents. Some email clients allow their functionality to be modified by users through the addition of add-ons. Protocols have also been defined for communicating between email clients and servers. Some clients support multiple protocols for doing the same task, allowing them to be configured according to email server specifications."

The Boxer application is an email application that is installed on mobile devices. This application allows access to enterprise email, calendar and contacts across both corporate-owned and employee-owned devices. Therefore, the conformance claim is appropriate.

3.8 Technical Decisions

Technical Decisions that effected the SFR wording have been annotated with a Footnote.

The following is a complete list of Technical Decisions that apply to the [APP_PP] and [EC_PP] evaluation activities that must be performed during the evaluation of this TOE:

TD #	Title	References	Changes			Analysis to this evaluation	
			SFR	AA	Notes	NA	Reason
TD0427	Reliable Time Source	A.Platform					Updated wording to Assumption.
TD0414*	FTP_ITC_EXT1. Tests 1 and 2	FTP_ITC_EXT.1		X			AA: Test wording
TD0405*	FIA_SASL_EXT.1 Testing	FIA_SASL_EXT.1		X		X	Not claiming FIA_SASL_EXT.1
TD0392	FCS_TLSC_EXT.1.2 Wildcard Checking	FCS_TLSC_EXT.1.2		X			AA: Test wording
TD0390	Cryptographically Secure RNG	FCS_RBG_EXT.1.1		X		X	Supersedes TD0172 Not claiming Windows
TD0389	Handling of SSH EP claim for platform	FTP_DIT_EXT.1	X	X	X		Supersedes TD0177 AA: TSS wording
TD0385	FTP_DIT_EXT.1 Assurance Activity Clarification	FTP_DIT_EXT.1		X		X	Not claiming VPN or IPSEC.
TD0382	Configuration Storage Options for Apps	FMT_MEC_EXT.1		X		X	Not claiming Windows or Linux
TD0381*	FCS_SMIME_EXT.1 Test 3	FCS_SMIME_EXT.1.1		X		X	Not claiming Windows or Linux
TD0380	Linux Keyring Requirement in FCS_STO_EXT.1	FCS_STO_EXT.1			X		Supersedes TD0192

TD0364	Android mmap testing for FPT_AEX_EXT.1.1	FCS_AEX_EXT.1.1		X			AA: Test wording
TD0359	Buffer Protection	FPT_AEX_EXT.1.5		X		X	Not claiming Windows
TD0358	Cipher Suites for TLS in SWApp v1.2	FCS_TLSC_EXT.1 and FCS_TLSS_EXT.1	X		X		Supersedes TD0283
TD0352*	Added key destruction options	FCS_CKM_EXT.4	X	X			AA: TSS wording
TD0327	Default file permissions for FMT_CFG_EXT.1.2	FMT_CFG_EXT.1.2	X	X	X		AA: Test wording
TD0326	RSA-based key establishment schemes	FCS_CKM.1, FCS_CKM.2, FCS_TLSS_EXT.1.3	X		X		Supersedes TD0293
TD0305	Handling of TLS connections with and without mutual authentication	FCS_TLSC_EXT.2.1, FCS_TLSC_EXT.1.4, FCS_TLSC_EXT.4.1		X			AA: Test wording Only FCS_TLS_EXT.2.1 only applies App_PP V1.2
TD0304	Update to FCS_TLSC_EXT.1.2	FCS_TLSC_EXT.1.2		X			AA: Test wording
TD0300	Sensitive Data in FDP_DAR_EXT.1	FDP_DAR_EXT.1	X	X	X		AA: TSS wording
TD0296	Update to FCS_HTTPS_EXT.1.3	FCS_HTTPS_EXT.1.3	X	X	X	X	Not claiming HTTP
TD0295	Update to FPT_AEX_EXT.1.3 Assurance Activities	FPT_AEX_EXT.1.3		X			Supersedes TD0269 AA: Testing wording for Android and Windows.
TD0268	FMT_MEC_EXT.1 Clarification	FMT_MEC_EXT.1			X		
TD0267	TLSS testing - Empty Certificate Authorities list	FCS_TLSS_EXT.1.5, FCS_TLSS_EXT.2.4, FCS_TLSS_EXT.1.4		X		X	Not claiming TLSS
TD0266*	Password/passphrase min vs max value for FCS_CKM_EXT.5.1	FCS_CKM_EXT.5.1	X	X	X		AA: TSS, AGD, Test wording
TD0254*	Algorithms in FCS_SMIME_EXT.1.4	FCS_SMIME_EXT.1.4	X				
TD0244	FCS_TLSC_EXT - TLS Client Curves Allowed	FCS_TLSC_EXT.2.1, FCS_TLSC_EXT.4.1, FCS_TLSC_EXT.1.4, FCS_TLSC_EXT.2/WLAN	X	X	X		AA: Test wording Only FCS_TLS_EXT.4.1 applies to APP_PP v1.2
TD0241	Removal of Test 4.1 in FCS_TLSS_EXT.1.1	FCS_TLSS_EXT.1.1, PP_APP_v1.2		X		X	Not claiming TLSS
TD0238	User-modifiable files FPT_AEX_EXT.1.4	FPT_AEX_EXT.1.4		X			AA: Test wording
TD0221	FMT_SMF.1.1 - Assignments moved to Selections	FMT_SMF.1.1	X				Supersedes TD0122

TD0217	Compliance to RFC5759 and RFC5280 for using CRLs	FIA_X509_EXT.1.1	X				
TD0215	Update to FCS_HTTPS_EXT.1.2	FCS_HTTPS_EXT.1.2			X	X	Not claiming HTTPS
TD0178	Integrity for installation tests in AppSW PP	FPT_TUD_EXT.1.3			X		
TD0174	Optional Ciphersuites for TLS	FCS_TLSC_EXT.1.1, FCS_EAP-TLS_EXT.1.1,	X				
TD0163	Update to FCS_TLSC_EXT.1.1 Test 5.4 and FCS_TLSS_EXT.1.1 Test	FCS_TLSC_EXT.1.1, FCS_TLSS_EXT.1.1		X			AA: Testing
TD0131	Update to FCS_TLSS_EXT.1.1 Test 4.5		X		X	X	Not claiming TLSS
TD0121	FMT_MEC_EXT.1.1 Configuration Options	FMT_MEC_EXT.1.1			X	X	Not claiming SWFE EP conformance
TD0119	FCS_STO_EXT.1.1 in PP_APP_v1.2	FCS_STO_EXT.1.1	X	X	X		AA: TSS and Testing
TD0107	FCS_CKM - ANSI X9.31-1998, Section 4.1.for Cryptographic Key Generation		X		X	X	Not claiming this selection or SFR

Table 6: Technical Decisions

* Technical Decisions that apply to the Application Software Extended Package for Email Clients

4 Security Problem Definition

4.1 Threats

This section identifies the threats against the TOE. These threats have been taken from the [APP_PP] and [EC_EP].

Threat	Threat Definition
T.FLAWED_ADDON	Email client functionality can be extended with integration of third-party utilities and tools. This expanded set of capabilities is made possible via the use of add-ons. The tight integration between the basic email client code and the new capabilities that add-ons provide increases the risk that malefactors could inject serious flaws into the email client application, either maliciously by an attacker, or accidentally by a developer. These flaws enable undesirable behaviors including, but not limited to, allowing unauthorized access to sensitive information in the email client, unauthorized access to the device's file system, or even privilege escalation that enables unauthorized access to other applications or the operating system.
T.LOCAL_ATTACK	An attacker can act through unprivileged software on the same computing platform on which the application executes. Attackers may provide maliciously formatted input to the application in the form of files or other local communications.
T.NETWORK_ATTACK	An attacker is positioned on a communications channel or elsewhere on the network infrastructure. Attackers may engage in communications with the application software or alter communications between the application software and other endpoints in order to compromise it.
T.NETWORK_EAVESDROP	An attacker is positioned on a communications channel or elsewhere on the network infrastructure. Attackers may monitor and gain access to data exchanged between the application and other endpoints.
T.PHYSICAL_ACCESS	An attacker may try to access sensitive data at rest.

Table 7: TOE Threats

4.2 Organizational Security Policies

There are no Organizational Security Policies in the [APP_PP] or [EC_EP].

4.3 Assumptions

The specific conditions listed in this section are assumed to exist in the TOE’s Operational Environment. These assumptions have been taken from the [APP_PP] or [EC_EP].

Assumption	Assumption Definition
A.PLATFORM[1]	The TOE relies upon a trustworthy computing platform with a reliable time clock for its Execution. This includes the underlying platform and whatever runtime environment it provides to the TOE.
A.PROPER_ADMIN	The administrator of the application software is not careless, willfully negligent or hostile, and administers the software within compliance of the applied enterprise security policy.

[1] TD0427

A.PROPER_USER	The user of the application software is not willfully negligent or hostile, and uses the software in compliance with the applied enterprise security policy.

Table 8: TOE Assumptions

4.4 Security Objectives

This section identifies the security objectives of the TOE and its supporting environment. The security objectives identify the responsibilities of the TOE and its environment in meeting the security needs.

4.4.1 TOE Security Objectives

This section identifies the security objectives of the TOE. These objectives have been taken directly from the [APP_PP] or [EC_EP].

Objective	Objective Definition
O.ADDON_INTEGRITY	To address issues associated with malicious or flawed plugins or extensions, conformant email clients implement mechanisms to ensure their integrity. This includes verification at installation time and update.
O.INTEGRITY	Conformant TOEs ensure the integrity of their installation and update packages, and also leverage execution environment-based mitigations. Software is seldom if ever shipped without errors, and the ability to deploy patches and updates to fielded software with integrity is critical to enterprise network security. Processor manufacturers, compiler developers, execution environment vendors, and operating system vendors have developed execution environment-based mitigations that increase the cost to attackers by adding complexity to the task of compromising systems. Application software can often take advantage of these mechanisms by using APIs provided by the runtime environment or by enabling the mechanism through compiler or linker options.
O.MANAGEMENT	To facilitate management by users and the enterprise, conformant TOEs provide consistent and supported interfaces for their security-relevant configuration and maintenance. This includes the deployment of applications and application updates through the use of platform-supported deployment mechanisms and formats, as well as providing mechanisms for configuration. This also includes providing control to the user regarding disclosure of any PII.
O.PROTECTED_COMMS	To address both passive (eavesdropping) and active (packet Modification) network attack threats, conformant TOEs will use a trusted channel for sensitive data. Sensitive data includes cryptographic keys, passwords, and any other data specific to the application that should not be exposed outside of the application.
O.PROTECTED_STORAGE	To address the issue of loss of confidentiality of user data in the event of loss of physical control of the storage medium, conformant TOEs will use data-at-rest protection. This involves encrypting data and keys stored by the TOE in order to prevent unauthorized access to this data. This also includes unnecessary Network Communications whose consequence may be the loss of data.
O.QUALITY	To ensure quality of implementation, conformant TOEs leverage services and APIs provided by the runtime environment rather than implementing their own versions of these services and APIs. This is especially important for cryptographic services and other complex operations such as file and media parsing. Leveraging this platform behavior relies upon using only documented and supported APIs.

Table 9: TOE Objectives

4.4.2 Security Objectives for the Operational Environment

The TOE's operational environment must satisfy the following objectives:

Objective	Objective Definition
OE.PLATFORM	The TOE relies upon a trustworthy computing platform for its execution. This includes the underlying operating system and any discrete execution environment provided to the TOE.
OE.PROPER_ADMIN	The administrator of the application software is not careless, willfully negligent or hostile, and administers the software within compliance of the applied enterprise security policy.
OE.PROPER_USER	The user of the application software is not willfully negligent or hostile, and uses the software within compliance of the applied enterprise security policy.

Table 10: Operational Environment Objectives

4.5 Security Problem Definition Rationale

The assumptions, threats, OSPs, and objectives that are defined in this ST represent the assumptions, threats, OSPs, and objectives that are specified in the Protection Profiles to which the TOE claims conformance. The associated mappings of assumptions to environmental objectives, SFRs to TOE objectives, and OSPs and objectives to threats are therefore identical to the mappings that are specified in the claimed Protection Profiles.

5 Extended Components Definition

5.1 Extended Security Functional Requirements

The extended Security Functional Requirements that are claimed in this ST are taken directly from the PPs to which the ST and TOE claim conformance. These extended components are formally defined in the PPs in which their usage is required.

5.2 Extended Security Assurance Requirements

The extended Security Assurance Requirement that is claimed in this ST is taken directly from the PP to which the ST and TOE claim conformance. This extended component is formally defined in the PP in which its usage is required.

6 Security Functional Requirements

6.1 Conventions

The CC permits four functional component operations—assignment, refinement, selection, and iteration—to be performed on functional requirements. This ST will highlight the operations in the following manner:

- **Assignment:** allows the specification of an identified parameter. Indicated with *italicized text*.
- **Refinement:** allows the addition of details. Indicated with **bold text**.
- **Selection:** allows the specification of one or more elements from a list. Indicated with <u>underlined text</u>.
- **Iteration operation:** are identified with a number inside parentheses (e.g. "(1)")

When multiple operations are combined, such as an assignment that is provided as an option within a selection or refinement, a combination of the text formatting is used.

Text that is formatted in a claimed PP, such as if the PP's instantiation of the SFR has a refinement (bolded font), or a completed assignment (inside brackets), the formatting is not preserved when reproduced in this ST. Only the assignments and selections made by the ST author are within [brackets]. This is so that the reader can easily identify the operations that are performed by the ST author.

6.2 Security Functional Requirements Summary

The following tables list the SFRs claimed by the TOE per platform. SFRs that originate from the Application Software Protection Profile are denoted by a [APP_PP], SFRs that originated from the Email Client Extended Package are denoted by [EC_EP].

Class Name	Component Identification	Component Name
Cryptographic Support	[APP_PP] FCS_CKM_EXT.1	Cryptographic Key Generation Services
	[APP_PP] FCS_CKM.1	Cryptographic Asymmetric Key Generation
	[APP_PP] FCS_CKM.2	Cryptographic Key Establishment
	[EC_EP] FCS_CKM_EXT.3	Protection of Key and Key Material
	[EC_EP] FCS_CKM_EXT.4	Cryptographic Key Destruction
	[EC_EP] FCS_CKM_EXT.5	Cryptographic Key Derivation (Password/Passphrase Conditioning)
	[EC_EP] FCS_COP_EXT.2	Key Wrapping
	[EC_EP] FCS_IVG_EXT.1	Initialization Vector Generation
	[EC_EP] FCS_KYC_EXT.1	Key Chaining
	[APP_PP] FCS_RBG_EXT.1	Random Bit Generation Services
	[EC_EP] FCS_SMIME_EXT.1	Secure/Multipurpose Internet Mail Extension (S/MIME)
	[APP_PP] FCS_STO_EXT.1(1)	Storage of Credentials
	[APP_PP] FCS_STO_EXT.1(2)	Storage of Credentials (iOS Revocation)
	[APP_PP] FCS_TLSC_EXT.1(1)	TLS Client Protocol (iOS)
	[APP_PP] FCS_TLSC_EXT.2	TLS Client Protocol
	[APP_PP] FCS_TLSC_EXT.4	TLS Client Protocol
User Data Protection	[APP_PP] FDP_DAR_EXT.1	Encryption of Sensitive Application Data

Class Name	Component Identification	Component Name
	[APP_PP] FDP_DEC_EXT.1(1)	Access to Platform Resources (iOS)
	[APP_PP] FDP_NET_EXT.1	Network Communications
	[EC_EP] FDP_NOT_EXT.1	Notification of S/MIME Status
	[EC_EP] FDP_SMIME_EXT.1	S/MIME
Identification and Authentication	[APP_PP] FIA_X509_EXT.1	X.509 Certificate Validation
	[APP_PP] FIA_X509_EXT.2	X.509 Certificate Authentication
	[EC_EP] FIA_X509_EXT.3	X.509 Authentication and Encryption
Security Management	[APP_PP] FMT_CFG_EXT.1	Secure by Default Configuration
	[APP_PP] FMT_MEC_EXT.1	Supported Configuration Mechanism
	[EC_EP] FMT_MOF_EXT.1	Management of Functions Behavior
	[APP_PP] FMT_SMF.1	Specification of Management Functions
Privacy	[APP_PP] FPR_ANO_EXT.1	User Consent for Transmission of Personally Identifiable Information
Protection of the TSF	[APP_PP] FPT_AEX_EXT.1	Anti-Exploitation Capabilities
	[EC_EP] FPT_AON_EXT.1	Support for Only Trusted Add-ons
	[APP_PP] FPT_API_EXT.1	Use of Supported Services and APIs
	[APP_PP] FPT_LIB_EXT.1	Use of Third Party Libraries
	[APP_PP] FPT_TUD_EXT.1	Integrity for Installation and Update
Trusted Path/Channels	[APP_PP] FTP_DIT_EXT.1 (1)	Protection of Data in Transit (iOS)
	[EC_EP] FTP_ITC_EXT.1	Inter-TSF Trusted Channel

Table 11a: iOS Security Functional Requirements for the TOE

Class Name	Component Identification	Component Name
Cryptographic Support	[APP_PP] FCS_CKM_EXT.1	Cryptographic Key Generation Services
	[APP_PP] FCS_CKM.1	Cryptographic Asymmetric Key Generation
	[APP_PP] FCS_CKM.2	Cryptographic Key Establishment
	[EC_EP] FCS_CKM_EXT.3	Protection of Key and Key Material
	[EC_EP] FCS_CKM_EXT.4	Cryptographic Key Destruction
	[EC_EP] FCS_CKM_EXT.5	Cryptographic Key Derivation (Password/Passphrase Conditioning)
	[APP_PP] FCS_COP.1(1)	Cryptographic Operation – Encryption/Decryption (Android)
	[APP_PP] FCS_COP.1(2)	Cryptographic Operation – Hashing (Android)
	[APP_PP] FCS_COP.1(3)	Cryptographic Operation – Signing (Android)
	[APP_PP] FCS_COP.1(4)	Cryptographic Operation - Keyed-Hash Message Authentication (Android)
	[EC_EP] FCS_COP_EXT.2	Key Wrapping
	[EC_EP] FCS_IVG_EXT.1	Initialization Vector Generation
	[EC_EP] FCS_KYC_EXT.1	Key Chaining
	[APP_PP] FCS_RBG_EXT.1	Random Bit Generation Services
	[APP_PP] FCS_RBG_EXT.2	Random Bit Generation from Application
	[EC_EP] FCS_SMIME_EXT.1	Secure/Multipurpose Internet Mail Extension (S/MIME)
	[APP_PP] FCS_STO_EXT.1(1)	Storage of Credentials
	[APP_PP] FCS_STO_EXT.1(3)	Storage of Credentials (Android Revocation)

Class Name	Component Identification	Component Name
	[APP_PP] FCS_TLSC_EXT.1(2)	TLS Client Protocol (Android)
	[APP_PP] FCS_TLSC_EXT.2	TLS Client Protocol
	[APP_PP] FCS_TLSC_EXT.4	TLS Client Protocol
User Data Protection	[APP_PP] FDP_DAR_EXT.1	Encryption of Sensitive Application Data
	[APP_PP] FDP_DEC_EXT.1(2)	Access to Platform Resources (Android)
	[APP_PP] FDP_NET_EXT.1	Network Communications
	[EC_EP] FDP_NOT_EXT.1	Notification of S/MIME Status
	[EC_EP] FDP_SMIME_EXT.1	S/MIME
Identification and Authentication	[APP_PP] FIA_X509_EXT.1	X.509 Certificate Validation
	[APP_PP] FIA_X509_EXT.2	X.509 Certificate Authentication
	[EC_EP] FIA_X509_EXT.3	X.509 Authentication and Encryption
Security Management	[APP_PP] FMT_CFG_EXT.1	Secure by Default Configuration
	[APP_PP] FMT_MEC_EXT.1	Supported Configuration Mechanism
	[EC_EP] FMT_MOF_EXT.1	Management of Functions Behavior
	[APP_PP] FMT_SMF.1	Specification of Management Functions
Privacy	[APP_PP] FPR_ANO_EXT.1	User Consent for Transmission of Personally Identifiable Information
Protection of the TSF	[APP_PP] FPT_AEX_EXT.1	Anti-Exploitation Capabilities
	[EC_EP] FPT_AON_EXT.1	Support for Only Trusted Add-ons
	[APP_PP] FPT_API_EXT.1	Use of Supported Services and APIs
	[APP_PP] FPT_LIB_EXT.1	Use of Third Party Libraries
	[APP_PP] FPT_TUD_EXT.1	Integrity for Installation and Update
	[APP_PP] FTP_DIT_EXT.1 (2)	Protection of Data in Transit (Android)
	[EC_EP] FTP_ITC_EXT.1	Inter-TSF Trusted Channel

Table 12b: Android Security Functional Requirements for the TOE

6.3 Security Functional Requirements

6.3.1 Class FCS: Cryptographic Support

6.3.1.1 *[APP_PP] FCS_CKM_EXT.1 Cryptographic Key Generation Services*

FCS_CKM_EXT.1.1

The application shall [invoke platform-provided functionality for asymmetric key generation].

6.3.1.2 *FCS_CKM.1(1) Cryptographic Asymmetric Key Generation*

FCS_CKM.1.1(1)[2]

The application shall [invoke platform-provided functionality] to generate asymmetric cryptographic keys in accordance with a specified cryptographic key generation algorithm [

RSA schemes using cryptographic key sizes of 2048-bit or greater that meet FIPS PUB 186-4, "Digital Signature Standard (DSS)", Appendix B.3;

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

] .

6.3.1.3 *[APP_PP] FCS_CKM.2 Cryptographic Key Establishment*

FCS_CKM.2.1[3]

The application shall [invoke platform-provided functionality] to perform Cryptographic Key Establishment in accordance with a specified Cryptographic Key Establishment method:

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

Elliptic curve-based key establishment schemes that meets the following: NIST Special Publication 800-56A, "Recommendation for Pair-Wise Key Establishment Schemes Using Discrete Logarithm Cryptography"].

6.3.1.4 *[EC_EP] FCS_CKM_EXT.3 Protection of Key and Key Material*

FCS_CKM_EXT.3.1

The email client shall [only store keys in non-volatile memory when wrapped as specified in FCS_COP_EXT.2 unless the key meets any one of following criteria:

[The plaintext key is the public portion of the key pair]].

6.3.1.5 *[EC_EP] FCS_CKM_EXT.4 Cryptographic Key Destruction*

FCS_CKM_EXT.4.1[4]

The email client shall [

- invoke platform-provided key destruction,
- implement key destruction using [

[2] TD0326

[3] TD0326

[4] TD0352

- For volatile memory, the erasure shall be executed by a [
  - single direct overwrite [
    - consisting of zeroes]].
- For nonvolatile storage, the erasure shall be executed by [
  - single]

  overwrite of key data storage location consisting of [
    - a static pattern]]]

that meet the following: [

- NIST SP800-88]

for destroying all keying material and cryptographic security parameters when no longer needed.

6.3.1.6 *[EC_EP] FCS_CKM_EXT.5 Cryptographic Key Derivation (Password/Passphrase Conditioning)*

FCS_CKM_EXT.5.1[5]

The TSF shall support a password/passphrase of up to [*512*] characters used to generate a password authorization factor.

FCS_CKM_EXT.5.2[6]

The TSF shall allow passwords to be composed of any combination of upper case characters, lower case characters, numbers, and the following special characters: "!", "@", "#", "$", "%", "^", "&", "*", "(", and ")", and [no other characters].

FCS_CKM_EXT.5.3[7]

The TSF shall perform Password-based Key Derivation Functions in accordance with a specified cryptographic algorithm HMAC- [SHA-256]], with [*10k for iOS and 20k for Android*] iterations, and output cryptographic key sizes [256] bits that meet the following: NIST SP 800-132.

FCS_CKM_EXT.5.4[8]

The TSF shall not accept passwords less than [a value settable by the administrator] and greater than the maximum password length defined in FCS_CKM_EXT.5.1.

6.3.1.7 *[APP_PP] FCS_COP.1(1) Cryptographic Operation – Encryption/Decryption (Android)*

FCS_COP.1.1(1)

The application shall perform encryption/decryption in accordance with a specified cryptographic algorithm

- AES-CBC (as defined in NIST SP 800-38A) mode;

[5] TD0266
[6] TD0266
[7] TD0266
[8] TD0266

and [no other modes] and cryptographic key sizes 256-bit and [128-bit].

6.3.1.8 *[APP_PP] FCS_COP.1(2) Cryptographic Operation - Hashing (Android)*

FCS_COP.1.1(2)

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

6.3.1.9 *[APP_PP] FCS_COP.1(3) Cryptographic Operation - Signing (Android)*

FCS_COP.1.1(3)

The application shall perform cryptographic signature services (generation and verification) in accordance with a specified cryptographic algorithm

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

6.3.1.10 *[APP_PP] FCS_COP.1(4) Cryptographic Operation - Keyed-Hash Message Authentication (Android)*

FCS_COP.1.1(4)

The application shall perform keyed-hash message authentication in accordance with a specified cryptographic algorithm

- HMAC-SHA-256 and

[no other algorithms] with key sizes [*256 bits*] and message digest sizes 256 and [no other size] bits that meet the following: FIPS Pub 198-1 The Keyed-Hash Message Authentication Code and FIPS Pub 180-4 Secure Hash Standard.

6.3.1.11 *[EC_EP] FCS_COP_EXT.2(1) Key Wrapping (iOS)*

FCS_COP_EXT.2.1(1)

The email client shall

[use platform-provided functionality to perform Key Wrapping]

in accordance with a specified cryptographic algorithm

[AES Key Wrap]

and the cryptographic key size

[256 bits (AES)]

that meet the following: ["NIST SP 800-38F" for Key Wrap (section 6.2) and Key Wrap with Padding (section 6.3)].

6.3.1.12 *[EC_EP] FCS_COP_EXT.2(2) Key Wrapping (Android)*

FCS_COP_EXT.2.1(2)

The email client shall

[implement functionality to perform Key Wrapping]

in accordance with a specified cryptographic algorithm

[AES Key Wrap]

and the cryptographic key size

[256 bits (AES)]

that meet the following: ["NIST SP 800-38F" for Key Wrap (section 6.2) and Key Wrap with Padding (section 6.3)].

6.3.1.13 *[EC_EP] FCS_IVG_EXT.1 Initialization Vector Generation*

FCS_IVG_EXT.1.1

The email client shall create IVs in the following manner: [CBC: IVs shall be non-repeating].

6.3.1.14 *[EC_EP] FCS_KYC_EXT.1 Key Chaining*

FCS_KYC_EXT.1.1

The email client shall maintain a key chain of:

[intermediate keys originating from:

[a password as specified in FCS_CKM_EXT.5.1]

to the data encryption/decryption key(s) using the following method(s):

implement Key Wrapping as specified in FCS_COP_EXT.2]

while maintaining an effective strength of [256 bits]].

6.3.1.15 *[APP_PP] FCS_RBG_EXT.1 Random Bit Generation Services*

FCS_RBG_EXT.1.1

The application shall [invoke platform-provided DRBG functionality, implement DRBG functionality] for its cryptographic operations.

6.3.1.16 *[APP_PP] FCS_RBG_EXT.2 Random Bit Generation from Application (Android)*

FCS_RBG_EXT.2.1

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

FCS_RBG_EXT.2.2

The deterministic RBG shall be seeded by an entropy source that accumulates entropy from a platform-based DRBG and [no other noise source] with a minimum of [256 bits] of entropy at least equal to the greatest security strength (according to NIST SP 800-57) of the keys and hashes that it will generate.

6.3.1.17 *[EC_EP] FCS_SMIME_EXT.1 Secure/Multipurpose Internet Mail Extension (S/MIME)*

FCS_SMIME_EXT.1.1

The email client shall implement both a sending and receiving S/MIME v3.2 Agent as defined in RFC 5751, using CMS as defined in RFCs 5652, 5754, and 3565.

FCS_SMIME_EXT.1.2

The email client shall transmit the ContentEncryptionAlgorithmIdentifier for AES-128 CBC and AES-256 CBC as part of the S/MIME protocol.

FCS_SMIME_EXT.1.3

The email client shall present the digestAlgorithm field with the following Message Digest Algorithm identifiers [id-sha256, id-sha384, id-sha512] and no others as part of the S/MIME protocol.

FCS_SMIME_EXT.1.4[9]

The email client shall present the signatureAlgorithm field with the following sha256withRSAEncryption and [no other algorithms] as part of the S/MIME protocol.

FCS_SMIME_EXT.1.5

The email client shall support use of different private keys (and associated certificates) for signature and for encryption as part of the S/MIME protocol.

FCS_SMIME_EXT.1.6

The email client shall only accept a signature from a certificate with the digitalSignature bit set as part of the S/MIME protocol.

FCS_SMIME_EXT.1.7

The email client shall implement mechanisms to retrieve certificates and certificate revocation information [[*at a frequency equal to the value received from OCSP responder or an administratively set value which overrides the value from OCSP responder or automatically retrieve if no previous OCSP responder value or set value exists*]] as part of the S/MIME protocol.

6.3.1.18 *[APP_PP] FCS_STO_EXT.1(1) Storage of Credentials*

FCS_STO_EXT.1.1(1)[10]

[9] TD0254

[10] TD0119

The application shall

[invoke the functionality provided by the platform to securely store [*iOS credentials defined in Table 15*],

implement functionality to securely store [*Android credentials defined in Table 14*]] to nonvolatile memory.

6.3.1.19 *[APP_PP] FCS_STO_EXT.1(2) Storage of Credentials (iOS Revocation)*

FCS_STO_EXT.1.1(2)[11]

The application shall

[invoke the functionality provided by the platform to securely store [*Server revocation credential information*],

implement functionality to securely store [*S/MIME revocation credential information*]] to nonvolatile memory.

6.3.1.20 *[APP_PP] FCS_STO_EXT.1(3) Storage of Credentials (Android Revocation)*

FCS_STO_EXT.1.1(3)[12]

The application shall

[invoke the functionality provided by the platform to securely store [*S/MIME and Server revocation credential information*] to nonvolatile memory.

6.3.1.21 *[APP_PP] FCS_TLSC_EXT.1(1) TLS Client Protocol (iOS)*

FCS_TLSC_EXT.1.1(1)[13]

The application shall [invoke platform-provided TLS 1.2] supporting the following ciphersuites:

TLS_RSA_WITH_AES_128_CBC_SHA as defined in RFC 5246,
TLS_RSA_WITH_AES_256_CBC_SHA256 as defined in RFC 5246,
TLS_ECDHE_ECDSA_WITH_AES_128_GCM_SHA256 as defined in RFC 5289,
TLS_ECDHE_ECDSA_WITH_AES_256_GCM_SHA384 as defined in RFC 5289,
TLS_ECDHE_RSA_WITH_AES_128_GCM_SHA256 as defined in RFC 5289,
TLS_ECDHE_RSA_WITH_AES_256_GCM_SHA384 as defined in RFC 5289].

FCS_TLSC_EXT.1.2(1)

The application shall verify that the presented identifier matches the reference identifier according to RFC 6125.

FCS_TLSC_EXT.1.3(1)

[11] TD0119
[12] TD0119
[13] TD0358

The application shall only establish a trusted channel if the peer certificate is valid.

6.3.1.22 *[APP_PP] FCS_TLSC_EXT.1(2) TLS Client Protocol (Android)*

FCS_TLSC_EXT.1.1(2)[14]

The application shall [invoke platform-provided TLS 1.2] supporting the following ciphersuites:

TLS_RSA_WITH_AES_128_CBC_SHA as defined in RFC 5246,
TLS_ECDHE_ECDSA_WITH_AES_128_GCM_SHA256 as defined in RFC 5289,
TLS_ECDHE_ECDSA_WITH_AES_256_GCM_SHA384 as defined in RFC 5289,
TLS_ECDHE_RSA_WITH_AES_128_GCM_SHA256 as defined in RFC 5289,
TLS_ECDHE_RSA_WITH_AES_256_GCM_SHA384 as defined in RFC 5289].

FCS_TLSC_EXT.1.2(2)

The application shall verify that the presented identifier matches the reference identifier according to RFC 6125.

FCS_TLSC_EXT.1.3(2)

The application shall only establish a trusted channel if the peer certificate is valid.

6.3.1.23 *[APP_PP] FCS_TLSC_EXT.2 TLS Client Protocol*

FCS_TLSC_EXT.2.1

The application shall support mutual authentication using X.509v3 certificates.

6.3.1.24 *[APP_PP] FCS_TLSC_EXT.4 TLS Client Protocol*

FCS_TLSC_EXT.4.1[15]

The application shall present the supported Elliptic Curves Extension in the Client Hello with the following NIST curves: [secp256r1, secp384r1].

6.3.2 Class FDP: User Data Protection

6.3.2.1 *[APP_PP] FDP_DAR_EXT.1 Encryption of Sensitive Application Data*

FDP_DAR_EXT.1[16]

The application shall [leverage platform-provided functionality to encrypt sensitive data] in non-volatile memory.

6.3.2.2 *[APP_PP] FDP_DEC_EXT.1(1) Access to Platform Resources (iOS)*

FDP_DEC_EXT.1.1(1)

[14] TD0358
[15] TD0244
[16] TD0300

The application shall restrict its access to [network connectivity, camera, [*device storage, phone, and touch/face ID*]].

FDP_DEC_EXT.1.2(1)

The application shall restrict its access to [address book, calendar].

6.3.2.3 *[APP_PP] FDP_DEC_EXT.1(2) Access to Platform Resources (Android)*

FDP_DEC_EXT.1.1(2)

The application shall restrict its access to [network connectivity, camera, NFC, [*device storage, phone, and fingerprint*]].

FDP_DEC_EXT.1.2(2)

The application shall restrict its access to [address book, calendar].

6.3.2.4 *[APP_PP] FDP_NET_EXT.1 Network Communications*

FDP_NET_EXT.1.1

The application shall restrict network communication to

[user-initiated communication for [*send email, force sync, GAL lookup, email search*],

respond to [*active sync (calendar, address book, email) via TCP port 443*]].

6.3.2.5 *[EC_EP] FDP_NOT_EXT.1 Notification of S/MIME Status*

FDP_NOT_EXT.1.1

The email client shall display a notification of the S/MIME status of received emails upon viewing.

6.3.2.6 *[EC_EP] FDP_SMIME_EXT.1 S/MIME*

FDP_SMIME_EXT.1.1

The email client shall use S/MIME to sign, verify, encrypt, and decrypt mail.

6.3.3 Class FIA: Identification and Authentication

6.3.3.1 *[APP_PP] FIA_X509_EXT.1 X.509 Certificate Validation*

FIA_X509_EXT.1.1

The application shall [implement functionality] to validate certificates in accordance with the following rules:

- RFC 5280 certificate validation and certificate path validation.
- The certificate path must terminate with a trusted CA certificate.

- The application shall validate a certificate path by ensuring the presence of the basicConstraints extension and that the CA flag is set to TRUE for all CA certificates.
- The application shall validate the revocation status of the certificate using [the Online Certificate Status Protocol (OCSP) as specified in RFC 2560].
- The application shall validate the extendedKeyUsage field according to the following rules:
    - Certificates used for trusted updates and executable code integrity verification shall have the Code Signing purpose (id-kp 3 with OID 1.3.6.1.5.5.7.3.3) in the extendedKeyUsage field.
    - Server certificates presented for TLS shall have the Server Authentication purpose (id-kp 1 with OID 1.3.6.1.5.5.7.3.1) in the extendedKeyUsage field.
    - Client certificates presented for TLS shall have the Client Authentication purpose (id-kp 2 with OID 1.3.6.1.5.5.7.3.2) in the extendedKeyUsage field.
    - S/MIME certificates presented for email encryption and signature shall have the Email Protection purpose (id-kp 4 with OID 1.3.6.1.5.5.7.3.4) in the extendedKeyUsage field.
    - OCSP certificates presented for OCSP responses shall have the OCSP Signing purpose (id-kp 9 with OID 1.3.6.1.5.5.7.3.9) in the extendedKeyUsage field.
    - Server certificates presented for EST shall have the CMC Registration Authority (RA) purpose (id-kp-cmcRA with OID 1.3.6.1.5.5.7.3.28) in the extendedKeyUsage field.

FIA_X509_EXT.1.2

The application shall only treat a certificate as a CA certificate if the basicConstraints extension is present and the CA flag is set to TRUE.

6.3.3.2 *[APP_PP] FIA_X509_EXT.2 X.509 Certificate Authentication*

FIA_X509_EXT.2.1

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

FIA_X509_EXT.2.2

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

6.3.3.3 *[EC_EP] FIA_X509_EXT.3 X.509 Authentication and Encryption*

FIA_X509_EXT.3.1

The email client shall use X.509v3 certificates as defined by RFC 5280 to support encryption and authentication for S/MIME.

FIA_X509_EXT.3.2

The email client shall prevent the establishment of a trusted communication channel when the peer certificate is deemed invalid.

FIA_X509_EXT.3.3

The email client shall prevent the installation of code if the code signing certificate is deemed invalid.

FIA_X509_EXT.3.4

The email client shall prevent the encryption of email if the email protection certificate is deemed invalid.

FIA_X509_EXT.3.5

The email client shall prevent the signing of email if the email protection certificate is deemed invalid.

6.3.4 Class FMT: Security Management

6.3.4.1 *[APP_PP] FMT_CFG_EXT.1 Secure by Default Configuration*

FMT_CFG_EXT.1.1

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

FMT_CFG_EXT.1.2[17]

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

6.3.4.2 *[APP_PP] FMT_MEC_EXT.1 Supported Configuration Mechanism*

FMT_MEC_EXT.1.1

The application shall invoke the mechanisms recommended by the platform vendor for storing and setting configuration options.

6.3.4.3 *[EC_EP] FMT_MOF_EXT.1 Management of Functions Behavior*

FMT_MOF_EXT.1.1

The email client shall be capable of performing the following management functions, controlled by the user or administrator as shown:

X: Mandatory O: Optional

Management Function	Administrator	User
Enable/disable plaintext only mode globally and by [no other method]	O	
Configure message sending/receiving to only use cryptographic algorithms defined in FCS_SMIME_EXT.1	O	
Change password/passphrase authentication credential		O

[17] TD0327

Configure cryptographic functionality	O
[*Configure Password Complexity Policy: Length*	O
Configure OCSP retrieval frequency]	O

6.3.4.4 *[APP_PP] FMT_SMF.1 Specification of Management Functions*

FMT_SMF.1.1

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

Change password/passphrase authentication credential

]].

6.3.5 Class FPR: Privacy

6.3.5.1 *[APP_PP] FPR_ANO_EXT.1 User Consent for Transmission of Personally Identifiable Information*

FPR_ANO_EXT.1.1

The application shall [not transmit PII over a network].

6.3.6 Class FPT: Protection of the TSF

6.3.6.1 *[APP_PP] FPT_AEX_EXT.1 Anti-Exploitation Capabilities*

FPT_AEX_EXT.1.1

The application shall not request to map memory at an explicit address except for [*the designated use of mmap for the use of "reallocating" memory to expand the Boxer database file map*].

FPT_AEX_EXT.1.2

The application shall [not allocate any memory region with both write and execute permissions].

FPT_AEX_EXT.1.3

The application shall be compatible with security features provided by the platform vendor.

FPT_AEX_EXT.1.4

The application shall not write user-modifiable files to directories that contain executable files unless explicitly directed by the user to do so.

FPT_AEX_EXT.1.5

The application shall be compiled with stack-based buffer overflow protection enabled.

6.3.6.2 *[EC_EP] FPT_AON_EXT.1 Support for Only Trusted Add-ons*

FPT_AON_EXT.1.1

The email client shall include the capability to load [no add-ons].

6.3.6.3 *[APP_PP] FPT_API_EXT.1 Use of Supported Services and APIs*

FPT_API_EXT.1.1

The application shall only use only documented platform APIs.

6.3.6.4 *[APP_PP] FPT_LIB_EXT.1 Use of Third Party Libraries*

FPT_LIB_EXT.1.1

The application shall be packaged with only [*see Android and iOS list details in Section 8.6.4*]

6.3.6.5 *[APP_PP] FPT_TUD_EXT.1 Integrity for Installation and Update*

FPT_TUD_EXT.1.1

The application shall [leverage platform] to check for updates and patches to the application software.

FPT_TUD_EXT.1.2

The application shall be distributed using the format of the platform-supported package manager.

FPT_TUD_EXT.1.3

The application shall be packaged such that its removal results in the deletion of all traces of the application, with the exception of configuration settings, output files, and audit/log events.

FPT_TUD_EXT.1.4

The application shall not download, modify, replace or update its own binary code.

FPT_TUD_EXT.1.5

The application shall [provide the ability] to query the current version of the application software.

FPT_TUD_EXT.1.6

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

6.3.7 Class FTP: Trusted Path/Channels

6.3.7.1 *[APP_PP] FTP_DIT_EXT.1(1) Protection of Data in Transit (iOS)*

FTP_DIT_EXT.1.1(1)[18]

The application shall [encrypt all transmitted sensitive data with [TLS]] between itself and another trusted IT product.

[18] TD0389

6.3.7.2 *[APP_PP] FTP_DIT_EXT.1(2) Protection of Data in Transit (Android)*

FTP_DIT_EXT.1.1(2)[19]

The application shall [encrypt all transmitted data with [TLS]] between itself and another trusted IT product.

6.3.7.3 *[EC_EP] FTP_ITC_EXT.1 Inter-TSF Trusted Channel*

FTP_ITC_EXT.1.1

The email client shall initiate or receive communication via the trusted channel.

FTP_ITC_EXT.1.2

The email client shall communicate via the trusted channel for [ActiveSync].

6.4 Statement of Security Functional Requirements Consistency

The Security Functional Requirements included in the ST represent all required SFRs specified in the PPs against which exact conformance is claimed a subset of the optional SFRs. All hierarchical relationships, dependencies, and unfulfilled dependency rationales in the ST are considered to be identical to those that are defined in the claimed PP.

[19] TD0389

7 Security Assurance Requirements

This section identifies the Security Assurance Requirements (SARs) that are claimed for the TOE. The SARs which are claimed are in exact conformance with the [APP_PP] and [EC_EP].

7.1 Class ASE: Security Target

As per ASE activities defined in [CEM]

7.2 Class ADV: Development

7.2.1 Basic Functional Specification (ADV_FSP.1)

7.2.1.1 *Developer action elements:*

ADV_FSP.1.1D

The developer shall provide a functional specification.

ADV_FSP.1.2D

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

7.2.1.2 *Content and presentation elements:*

ADV_FSP.1.1C

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

ADV_FSP.1.2C

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

ADV_FSP.1.3C

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

ADV_FSP.1.4C

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

7.2.1.3 *Evaluator action elements:*

ADV_ FSP.1.1E

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

ADV_ FSP.1.2E

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

7.3 Class AGD: Guidance Documentation

7.3.1 Operational User Guidance (AGD_OPE.1)

7.3.1.1 *Developer action elements:*

AGD_OPE.1.1D

The developer shall provide operational user guidance.

7.3.1.2 *Content and presentation elements:*

AGD_OPE.1.1C

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

AGD_OPE.1.2C

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

AGD_OPE.1.3C

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

AGD_OPE.1.4C

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

AGD_OPE.1.5C

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

AGD_OPE.1.6C

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

AGD_OPE.1.7C

The operational user guidance shall be clear and reasonable.

7.3.1.3 *Evaluator action elements:*

AGD_OPE.1.1E

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

7.3.2 Preparative Procedures (AGD_PRE.1)

7.3.2.1 *Developer action elements:*

AGD_PRE.1.1D

The developer shall provide the TOE including its preparative procedures.

7.3.2.2 *Content and presentation elements:*

AGD_ PRE.1.1C

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

AGD_ PRE.1.2C

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

7.3.2.3 *Evaluator action elements:*

AGD_ PRE.1.1E

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

AGD_ PRE.1.2E

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

7.4 Class ALC: Life Cycle Support

7.4.1 Labeling of the TOE (ALC_CMC.1)

7.4.1.1 *Developer action elements:*

ALC_CMC.1.1D

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

7.4.1.2 *Content and presentation elements:*

ALC_CMC.1.1C

The TOE shall be labeled with its unique reference.

7.4.1.3 *Evaluator action elements:*

ALC_CMC.1.1E

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

7.4.2 TOE CM Coverage (ALC_CMS.1)

7.4.2.1 *Developer action elements:*

ALC_CMS.1.1D

The developer shall provide a configuration list for the TOE.

7.4.2.2 *Content and presentation elements:*

ALC_CMS.1.1C

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

ALC_CMS.1.2C

The configuration list shall uniquely identify the configuration items.

7.4.2.3 *Evaluator action elements:*

ALC_CMS.1.1E

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

7.4.3 Timely Security Updates (ALC_TSU_EXT.1)

7.4.3.1 *Developer Actions Element:*

ALC_TSU_EXT.1.1D

The developer shall provide a description in the TSS of how timely security updates are made to the TOE.

ALC_TSU_EXT.1.2D

The developer shall provide a description in the TSS of how users are notified when updates change security properties or the configuration of the product.

7.4.3.2 *Content and presentation elements:*

ALC_TSU_EXT.1.1C

The description shall include the process for creating and deploying security updates for the TOE software.

ALC_TSU_EXT.1.1C

The description shall express the time window as the length of time, in days, between public disclosure of a vulnerability and the public availability of security updates to the TOE.

ALC_TSU_EXT.1.1C

The description shall include the mechanisms publicly available for reporting security issues pertaining to the TOE.

7.4.3.3 *Evaluator action elements:*

ALC_TSU_EXT.1.1E

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

7.5 Class ATE: Tests

7.5.1 Independent Testing - Conformance (ATE_IND.1)

7.5.1.1 *Developer action elements:*

ATE_IND.1.1D

The developer shall provide the TOE for testing.

7.5.1.2 *Content and presentation elements:*

ATE_IND.1.1C

The TOE shall be suitable for testing.

7.5.1.3 *Evaluator action elements:*

ATE_IND.1.1E

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

ATE_IND.1.2E

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

7.6 Class AVA: Vulnerability Assessment

7.6.1 Vulnerability Survey (AVA_VAN.1)

7.6.1.1 *Developer action elements:*

AVA_VAN.1.1D

The developer shall provide the TOE for testing.

7.6.1.2 *Content and presentation elements:*

AVA_VAN.1.1C

The TOE shall be suitable for testing.

7.6.1.3 *Evaluator action elements:*

AVA_VAN.1.1E

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

AVA_VAN.1.2E

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

AVA_VAN.1.3E

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

8 TOE Summary Specification

The following sections identify the security functions of the TOE and describe how the TSF meets each claimed SFR. They include Cryptographic Support, User Data Protection, Identification and Authentication, Security Management, Privacy, Protection of the TSF, and Trusted Path/Channels.

8.1 Cryptographic Support

[iOS] TLS and S/MIME cryptographic services for the Boxer application are provided by the underlying platform when the application is installed on an iPhone 8 running the iOS 11. In the evaluated configuration, the TOE is installed on Apple iOS 11 that uses the Apple iOS CoreCrypto Module v8.0. The specific cryptographic implementation for the iOS platform can be found in the Apple iOS 11 Security Target documentation (VID10851).

[Android] TLS communication cryptographic services for the Boxer application are provided by the underlying platform when the application is installed on an Samsung Galaxy S8+ mobile device running Android 8.In the evaluated configuration, the TOE is installed on Samsung Galaxy S8+ mobile device running Android 8 uses the Samsung BoringSSL Cryptographic Module. The specific cryptographic implementation for the Android platform can be found in the Samsung Android 8.0 Security Target documentation (VID10898).

Additionally, when Boxer is installed on a device running the Android OS 8.0, the application includes OpenSSL software library 1.0.2.p to perform the cryptographic functionality for S/MIME functionality. The CAVP certificates for Boxer’s OpenSSL Android implementation are specified in Table 12.

OpenSSL Algorithm for S/MIME	SFR	Consolidated CAVP Cert. #
HMAC-SHA-256, 256 bit key	FCS_CKM_EXT.5.3, FCS_COP.1(4)	C631
AES-128-CBC and AES-256-CBC	FCS_SMIME_EXT.1.2, FCS_COP.1(1)	C631
SHA-256, SHA-384, SHA-512	FCS_SMIME_EXT.1.3, FCS_COP.1(2)	C631
RSA (2048, SHA2-256)	FCS_SMIME_EXT.1.4, FCS_COP.1(3)	C631
DRBG CTR (AES-256)	FCS_RBG_EXT.1.1	C631
AES-256-CBC	FCS_COP.1(1) - Encryption of Boxer specific database used in support of FCS_STO_EXT.1(1) storage of specific keys.	C631

Table 13: OpenSSL CAVP Certificates

Below is a table that references which library is used according to the cryptographic function for the Android OS.

SFR	Cryptographic Function	iOS	Android	
		Platform	Platform	OpenSSL
[APP_PP] FCS_CKM_EXT.1	Cryptographic Key Generation Services	X	X	N/A
[APP_PP] FCS_CKM.1	Cryptographic Key Establishment	X	X	N/A
[APP_PP] FCS_CKM.2	Cryptographic Key Establishment	X	X	N/A
[APP_PP] FCS_COP.1(1)	Cryptographic Operation – Encryption/Decryption	X	X	X
[APP_PP] FCS_COP.1(2)	Cryptographic Operation - Hashing	X	X	X
[APP_PP] FCS_COP.1(3)	Cryptographic Operation - Signing	X	X	X
[APP_PP] FCS_COP.1(4)	Cryptographic Operation - Keyed-Hash Message Authentication	X	X	X
[APP_PP] FCS_RBG_EXT.1	Random Bit Generation Services	X	X	N/A
[APP_PP] FCS_RBG_EXT.2	Random Bit Generation from Application	N/A	N/A	X
[APP_PP] FCS_STO_EXT.1(1)	Storage of Credentials	X	X	X
[APP_PP] FCS_STO_EXT.1(2)	Storage of Credentials (iOS Revocation)	X	N/A	N/A
[APP_PP] FCS_STO_EXT.1(2)	Storage of Credentials (Android Revocation)	N/A	X	X
[APP_PP] FCS_TLSC_EXT.1(1)	TLS Client Protocol (iOS)	X	N/A	N/A
[APP_PP] FCS_TLSC_EXT.1(2)	TLS Client Protocol (Android)	N/A	X	N/A
[APP_PP] FCS_TLSC_EXT.2	TLS Client Protocol	X	X	N/A
[APP_PP] FCS_TLSC_EXT.4	TLS Client Protocol	X	X	N/A
[EC_EP] FCS_CKM_EXT.3	Protection of Key and Key Material	X	N/A	X
[EC_EP] FCS_CKM_EXT.4	Cryptographic Key Destruction	X	N/A	X
[EC_EP] FCS_CKM_EXT.5	Cryptographic Key Derivation (Password/Passphrase Conditioning)	X	N/A	X
[EC_EP] FCS_COP_EXT.2	Key Wrapping	X	N/A	X
[EC_EP] FCS_IVG_EXT.1	Initialization Vector Generation	X	N/A	X
[EC_EP] FCS_KYC_EXT.1	Key Chaining	X	N/A	X
[EC_EP] FCS_SMIME_EXT.1	Secure/Multipurpose Internet Mail Extension (S/MIME)	X	N/A	X

Table 14: Cryptographic Libraries

8.1.1 [APP_PP] FCS_CKM_EXT.1 and FCS_CKM.1.1(1)

The TOE uses pre-assigned certificates that are generated on the UEM (operational environment) prior to installation. The pre-assigned certificates are used for communications (provided by the TOE platform) between the Exchange server and the TOE and for user S/MIME functionality.

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

8.1.2 [APP_PP] FCS_CKM.2:

The TOE invokes the platform in support of two key establishment schemes for establishment of TLS communications.

- RSA key establishment conforming to “NIST SP 800-56B”.
- Elliptic curve-based key establishment conforming to NIST Special Publication 800-56A.

The TOE invokes the platform-provided functionality to perform Cryptographic Key Establishment for both Apple iOS 11 (VID10851) and Android 8.0 (VID10898).

8.1.3 [EC_EP] FCS_CKM_EXT.3:

Keys are wrapped according to the chain process described in FCS_COP_EXT.2 and FCS_KYC_EXT.1. The keys stored in nonvolatile memory are detailed in FCS_STO_EXT.1(1).

The SDK database and iOS keychain, where the Boxer keys are stored, are encrypted using AES-256-CBC.

> [Android] The SDK database where the Boxer keys are stored, is encrypted using AES-256-CBC. This cryptographic functionality is handled by the OpenSSL.
>
> [iOS] The iOS keychain, where the Boxer keys are stored, are encrypted using AES-256-CBC. This cryptographic functionality is handled by the iOS platform.

8.1.4 [EC_EP] FCS_CKM_EXT.4:

The TOE invokes and implements key destruction on both platforms. The method is dependent on the key and its storage location as to whether the TOE will implement or invoke this functionality.

When a plaintext or private keys stored in volatile or non-volatile memory is no longer needed, for example a TLS communication session closed, task completed, or the application being wiped, the TOE either implements or invokes the platform to perform the key destruction. Key destruction performed by Boxer is done with a single overwrite consisting of zeroes before releasing the volatile memory and a single overwrite consisting of a static pattern of zeroes before releasing the non-volatile storage space that meet NIST SP800-88. Key destruction performed by the OS is done in accordance with:

[Android] The specific cryptographic implementation for the Android platform can be found in the Samsung Android 8.0 Security Target documentation (VID10898).

[iOS] The specific cryptographic implementation for the iOS platform can be found in the Apple iOS 11 Security Target documentation (VID10851).

Table 14 and Table 15 describe the circumstances in which the key is no longer needed for volatile and nonvolatile memory. The tables also identify the mechanism (OS or Boxer) which is responsible for the destruction of the keys.

8.1.5 **[EC_EP] FCS_CKM_EXT.5:**

The TOE implements a password/passphrase for key chaining. The TOE supports a password of up to 512 characters. The supported character sets include: upper case, lower case, numeric, and only the following special characters "!", "@", "#", "$", "%", "^", "&", "*", "(", and ")". Passwords are not accepted if they are less than the required length configurable by the administrator or are greater than 512 characters.

The TOE performs a password-based key derivation function in accordance with the HMAC-SHA-256 algorithm, defined in COP.1(4)) with the password string without padding, a salt of 256 bits, 10K iterations (iOS) and 20K iterations (Android), with an output cryptographic key size of 256 bits that meets NIST SP 800-132.

8.1.6 **[APP_PP] FCS_COP.1(1):**

The TOE supports AES encryption/decryption for FCS_SMIME_EXT.1.2. AES-CBC is the supported mode with 256-bit and 128-bit keys supported.

[Android] OpenSSL provides the cryptographic support for S/MIME support and encryption/decryption services for the Boxer database using AES-256-CBC. This is in support of the storage of specific keys as defined in FCS_STO_EXT.1.

[iOS] The iOS platform provides the cryptographic support for both S/MIME support and encryption/decryption of the Boxer database using AES-256-CBC. This is in support of the storage of specific keys as defined in FCS_STO_EXT.1.

8.1.7 **[APP_PP] FCS_COP.1(2):**

The TOE supports the use of SHA-256, SHA-384, and SHA-512 for S/MIME functionality as defined in FCS_SMIME_EXT.1.3.

[Android] This cryptographic functionality is handled by the OpenSSL.

[iOS] This cryptographic functionality is handled by the iOS platform.

The following specifies the selected algorithms, key sizes, and message digest sizes respectively:

- SHA-256, 256, 256
- SHA-384, 384, 384
- SHA-512, 512, 512

8.1.8 [APP_PP] FCS_COP.1(3):

The TOE uses sha256withRSAEncryption for the signatureAlgorithm (RSA scheme) for digital signature services in support of S/MIME functionality as defined in FCS_SMIME_EXT.1.4. RSA scheme with a key size of 2048 bits is supported.

[Android] This cryptographic functionality is handled by the OpenSSL.

[iOS] This cryptographic functionality is handled by the iOS platform.

8.1.9 [APP_PP] FCS_COP.1(4):

The TOE performs a password-based key derivation function in accordance with the HMAC-SHA-256 algorithm with a 256 bit key size and 256 message digest size (defined by FCS_CKM_EXT.5.3).

[Android] This cryptographic functionality is handled by the OpenSSL.

[iOS] This cryptographic functionality is handled by the iOS platform.

8.1.10 [EC_EP] FCS_COP_EXT.2:

The keys are stored in a database encrypted using AES-256-CBC provided by OpenSSL (Android) and iOS CoreCrypto Module. Prior to their storage, the keys are wrapped as described below.

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

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

8.1.11 [EC_EP] FCS_IVG_EXT.1:

The TOE creates IVs using the CBC encryption mode meaning that the IVs are non-repeating.

8.1.12 [EC_EP] FCS_KYC_EXT.1:

For both Android and iOS, the user enters a password according to the policy set forth by FCS_CKM_EXT.5. This key is processed as prescribed by FCS_COP_EXT.2 and results in a 256-bit key (Password Key). The derived Password Key is only stored in volatile memory (RAM) and is always recreated when the user is required to enter the password to unlock the application such as first time executing the application after a reboot of the mobile device or restart of app. Users using the iOS may also input biometrics utilizing the Touch ID feature. A Touch ID key is randomly generated using iOS-platform provided functionality. The Key Chaining process is the same with the exception of using the Touch ID key instead of the Password Key.

Android Key Chaining Process:

When application is not running (i.e. first time login from a reboot):

- The user inputs the password
- The Passphrase Key is derived (SHA1 hashed and encoded version of the password)
- The encrypted Key Encryption Key (KEK1) is pulled from SDK *SecurePref* storage
- The encrypted Key Encryption Key is then decrypted using Passphrase Key

Or when session is running in background:

- Session Key in memory
- The encrypted Key Encryption Key (KEK2) is pulled from SDK *SecurePref* storage
- The encrypted Key Encryption Key is then decrypted using Session Key
- The encrypted Master key is pulled from SDK *SecurePref* storage
- The encrypted Master Key is then decrypted using Key Encryption Key
- The encrypted DB Key is then pulled from SDK *SecurePref* storage
- The encrypted DB Key is then decrypted using the Master Key
- Access to Boxer DB is now available

iOS Key Chaining Process:

- The user inputs password or biometric unlock
- The Password Key/Touch ID Key is derived
- The encrypted Master Key is pulled from iOS keychain
- The Master Key is then decrypted using Password Key/Touch ID key
- The encrypted SDK Boxer App Key is pulled from iOS keychain
- The encrypted SDK Boxer App Key is decrypted using Master Key
- The encrypted Boxer Master Key is then pulled form iOS SDK keychain
- The encrypted Boxer Master Key is decrypted using SDK Boxer App Key
- The encrypted Boxer DB key is pulled from iOS keychain
- The encrypted Boxer DB Key is then decrypted using Boxer Master Key
- Access to Boxer DB is now available

The TOE stores Boxer User key data (own certificate and those of S/MIME email) in an internal database which is fully encrypted. The TOE's internal database uses SQLCipher AES-256-CBC provided by OpenSSL (for Android) Apple iOS CoreCrypto Module for iOS.

8.1.13 [APP_PP] FCS_RBG_EXT.1 and [APP_PP] FCS_RBG_EXT.2:

[Android] When installed on an Android platform, the TOE implements its own DRBG functionality and invokes the platform's DRBG services, depending upon the function.

The TOE implements OpenSSL to provide NIST SP 800-90A compliant AES_CTR DRBG services (see Table 12 for certificate number) for the cryptographic functionality specified in the [EC_EP]. There is no ability to specify the use of an alternative DRBG. The TOE's DRBG is seeded with entropy data that is collected from /dev/random or /dev/urandom depending on the function. The amount of entropy that is collected is based on the function that the DRBG is being used for. In all cases, this amount is greater than or equal to the security strength of the data that is being outputted. The entropy source is described in greater detail in the proprietary Entropy Assessment Report.

The TOE invokes the platform's BoringSSL cryptography to provide NIST SP 800-90A compliant AES_CTR DRBG functionality for trusted communications between the TOE and email Exchange server. The specific cryptographic implementation for the Android platform can be found in the Samsung Android 8.0 Security Target documentation (VID10898). When the application invokes the platform to obtain random numbers, the platform APIs used are /dev/random and /dev/urandom APIs depending on function. The TOE also implements its own DRBG function using OpenSSL. The random numbers are obtained using /dev/random or /dev/urandom depending on what function when OpenSSL is being used.

[iOS] When installed on an iOS platform, the TOE invokes the 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. The CoreCrypto AES_CTR DRBG services also provide support for the cryptographic functionality specified in the [EC_EP]. The specific cryptographic implementation for the iOS platform can be found in the Apple iOS 11 Security Target documentation (VID10851). The platform API used to obtain random numbers is SecRandomCopyBytes().

8.1.14 [EC_EP] FCS_SMIME_EXT.1:

The TOE implements both a sending and receiving S/MIME v3.2 Agent as defined in RFC 5751, using CMS as defined in RFCs 5652, 5754, and 3565. The TOE uses AES-256-CBC (FCS_COP.1(1)) as its default for transmitting the ContentEncryptionAlgorithmIdentifier. The administrator is able to configure the TOE to use AES-128-CBC as well. For the digestAlgorithm, the TOE implements id-sha256 by default. This is also configurable by the administrator to use id-sha384 and id-sha512 (FCS_COP.1(2)). The TOE uses sha256withRSAEncryption for the signatureAlgorithm by default (FCS_COP.1(3). Only a certificate with the digitalSignature bit set will be accepted as part of the S/MIME protocol. The TOE supports the use of different private keys for signature and for encryption as part of the S/MIME protocol.

Certificate revocation information is received from the OCSP responder at a frequency defined by the server. An UEM administrator may override this frequency by configuring a value via UEM.

Certificates are pulled from the signed email when received in the application in order to validate the authenticity and integrity of the email.

8.1.15 [APP_PP] FCS_STO_EXT.1(1), [APP_PP] FCS_STO_EXT.1 (2) & [APP_PP] FCS_STO_EXT.1 (3)

The TOE invokes the platform and implements the storage of keys depending on the type of keys on both platforms. Keys are stored in the Boxer database that is encrypted using AES and the OS keychains. The following table includes the credentials that are stored in the encrypted database or keystore as well as their purpose as scoped by [APP_PP] and [EC_EP]:

Note: The term "SDK SecurePref" means a key grouping within the OS keychain.

Note: The term "App Wipe" is a remote wipe accomplished from the UEM console and "App Uninstall" is when a user removes it manually from the OS App listing.

Keys maintained by SDK (Directly/Indirectly use by Boxer)	Volatile	Non-volatile (db)/Keychain	Description
User entered Password/passcode	Used in memory Cleared by OS: • Device Reboot Zeroed by OS:	Not persistently stored	• User password entry

	- Functionality complete		
Passphrase Key	Used in memory Zeroed by Boxer: - Task completion	Not persistently stored	- This is a SHA1 hashed and encoded version of the key above (User entered Password/passcode)
Session Key	Used in memory Cleared by Boxer: - App Wipe Cleared by OS: - Device Reboot	Android: Stored in SharedPrefs and protected by the Android hardware-backed crypto services API keystore. Cleared by OS: - App Uninstall	- Primary key to be used to get to Key Encryption Key while the application is running
Key Encryption Key (KEK)	Used in memory Kept in Memory wrapped with Session Key (KEK2) until: Cleared by Boxer: - App Wipe	Wrapped using Passphrase Key and saved in SharedPrefs (KEK1) Wrapped using the Session Key and saved in SharedPrefs (KEK2) Cleared by OS: - App Uninstall	- Randomly generated at installation - KEK used to encrypt Master Key - The Key Encryption Key #1 and #2 is derived by encrypting the randomly generated Key Encryption Key with the Passphrase Key and Session Key respectively - When the mobile device user enters the correct passcode, then KEK1 can be decrypted to reproduce the KEK for use in volatile memory - When an active session, the KEK2 can be decrypted to reproduce the KEK for us in volatile memory
Master Key	Used in memory Cleared by OS: - Device Reboot	Wrapped using KEK and saved in SharedPrefs Cleared by OS: - App Uninstall	- Randomly generated at installation - Used to encrypt the Boxer Database Key
Key for Email Client (Email Client PP) (Used by Boxer exclusively)	**Volatile**	**Non-volatile (db)/Keychain**	**Description**
Boxer Database Key (Boxer DB Key)	Used in memory	Wrapped using Master Key and saved in SDK	- Randomly generated at installation

	Cleared by OS: - App Wipe - Device Reboot	SecurePref Cleared by OS: - App Uninstall AES wrapped with Master Key and stored in Android Shared preferences.	- Used to encrypt/decrypt Boxer DB providing access to mail data
Authentication Credential Password	Used in memory Cleared by OS: - App Wipe - Device Reboot	Persistently stored in AES-256-CBC encrypted Boxer Database Cleared by OS: - App Uninstall	- Used to authenticate boxer client with Exchange server
Authentication credential Communication Certificate	Used in memory Cleared by OS: - App Wipe - Device Reboot	Certificate is AES wrapped and stored in SDK SecurePref Cleared by OS: - App Uninstall Overwritten when a new certificate is generated	- Used for communication with Exchange server
S/MIME certificates containing public + private keys (1 for Encryption 1 for signing or the same certificate for both signing and encryption)	Used in memory Cleared by OS: - App Wipe - Device Reboot Encrypting/decrypting S/MIME email is completely done in memory	Certificate is stored in the SDK SecurePref. Cleared by OS: - App Uninstall A copy is also stored in the encrypted Boxer DB Cleared by OS: - App Uninstall	- Used for S/MIME
Public Key of email recipient	Used in memory Cleared by OS: - App Uninstall	Certificates are stored in the encrypted Boxer Database Cleared by OS: - App Uninstall Public Certificate and its Certificate chain are temporarily stored on the file system for the duration of the encryption/signature verification	- Used for S/MIME
Attachment	Used in memory	Encrypted version Stored in	- Used to encrypt attachments stored in the internal file

Encryption Key	Available only when accessing the attachment Zeroed by Boxer: • Task Completion (Decryption of attachment completed)	encrypted Boxer Database Cleared by OS: • App Uninstall	system
Temporary Files Encryption Key	Used in memory Available only when accessing the temporary file Zeroed by Boxer: • Task Completion (Temporary file no longer in use)	In case of “draft e-mails (before being sent out)”: Not stored persistently In case of “incoming S/MIME messages content”: Key is key-wrapped using Boxer database key and then encrypted using a static passcode and stored in Shared Preferences. Cleared by OS: • App Uninstall	• Used to store temporary content Example: Emails before being sent out utilize these files to temporarily store the content and are deleted once the message is sent.

Table 15: Stored Android Credentials

Keys maintained by SDK (Directly/Indirectly use by Boxer)	**Volatile**	**Non-volatile (db)/Keychain**	**Description**
Session Key	Used in memory Zeroed by Boxer: • Session Lock • App Wipe	Encrypted with App's public key (that was subscribed to share session) and saved in iOS Keychain Corresponding private key is kept in the running memory of the app Zeroed by Boxer: • Session lock • App Wipe	• Primary key to be used to get to Master Key while the application is running • Used to maintain app specific or shared session across SSO Apps
Biometric Key (i.e. TouchID Key)	Used in memory Not kept in Memory Zeroed by Boxer: • Session establishment completed	Wrapped by system and stored in Secure Enclave by system for type SecClass Key Zeroed by Boxer: • App Wipe • Passcode change	• Used only to unwrap Master Key and start creating a session
Password Based Key	Used in memory Not kept in Memory Zeroed by Boxer:	Not persistently stored	• Used only to unwrap Master Key and start creating a session • Uses KDF with 10000 iterations

	- Task Completion (Session establishment completed)		to convert user input into Password Based Key
Master Key	Used in memory Kept in Memory wrapped with Session Key (MK3) Zeroed by Boxer: - Session Lock - App Wipe	Wrapped with Password Based Key and saved in iOS keychain (MK1) Wrapped using Touch ID Key and saved in iOS keychain (MK2) Wrapped using Session Key and saved in iOS keychain (MK3) Zeroed by OS: - App Uninstall	- The Master Key #1, #2, #3 is derived by encrypting the randomly generated Master Key with the Password Key, Biometric Key, and Session key respectively - When the mobile device user enters the correct passcode, then MK1 can be decrypted to re-produce the Master Key for use in volatile memory - When the mobile device user enters the correct TouchID, the MK2 can be decrypted to re-produce the Master Key for use in volatile memory - When an active session, the MK3 can be decrypted to reproduce the Master Key for use in volatile memory
Shared Container Key	Used in memory Not Kept in Memory Zeroed by Boxer: - Task Completion	Wrapped with Master key and stored in iOS Keychain Destroyed by Boxer: - App Wipe - Logout	- Similar to App Key but this key is shared across the application in shared group
SDK Boxer App Key	Used in memory Obfuscated with Master Key which is also kept in memory Zeroed by Boxer: - Session Lock - App Wipe	Wrapped with Master key and stored in iOS Keychain Destroyed by Boxer: - App Wipe	- The Boxer App Key is randomly generated - The Boxer App Key is encrypted with the Master Key for storage - Used for encrypting/decrypting Boxer DB

Email Client Keys (Email Client PP) **(Used by boxer exclusively)**	**Volatile**	**Non-volatile (db)/Keychain**	**Description**
Boxer Master Key	Used in memory throughout the lifetime of the app	Stored encrypted in the Keychain Encrypted using the SDK	- Randomly generated at installation - Used for encrypting all other keys inside the boxer app

	Zeroed by OS: - App Close	Application Key Zeroed by OS: - App Uninstall	
Boxer Database Key	Used in-memory Zeroed by Boxer: - Task Completion (Opening Boxer Database)	Stored encrypted in the Keychain Encrypted with the Boxer Master Key Zeroed by OS: - App Uninstall	- Randomly generated at installation - Used for locking / unlocking the Boxer Database
File Encryption Keys	Used in-memory Zeroed by Boxer: - Task Completion (File Read or Write)	Stored encrypted in the Boxer Database Encrypted with the Boxer Master Key Zeroed by OS: - App Uninstall	- Used for encrypting / decrypting files on disk
S/MIME Private Keys	Used in-memory Loaded on demand Zeroed by OS: - App Close	Stored encrypted in the Boxer Database Encrypted with the Boxer Master Key Zeroed by OS: - App Uninstall	- Used for encrypting / decrypting S/MIME email messages
Exchange CBA Certificate	Used in memory throughout the lifetime of the app Zeroed by OS: - App Close	Stored encrypted in the Boxer Database Zeroed by Boxer: - App Wipe Zeroed by OS: - App Uninstall	- Used for Client-Server authentication with Exchange
Exchange Basic Authentication Password	Used in memory throughout the lifetime of the app Zeroed by OS: - App Close	Stored encrypted in the Boxer Database Zeroed by Boxer: - App Wipe Zeroed by OS: - App Uninstall	- Used for Client-Server authentication with Exchange

Table 16: Stored iOS Credentials

The Boxer application is designed to store certificate revocation status information. The following information is stored:

- Certificate Identifier
- Certificate Revocation Status
- Validation Failure Reason
- Next Revocation Check Date
- Last Revocation Checked Date

The information is overwritten when information is refreshed and not deleted. The revocation status information is deleted upon the Boxer application removal.

[iOS] The Boxer application implements the functionality to securely store S/MIME certificate revocation status information in the encrypted Boxer database. The iOS platform is invoked to securely store server certificate revocation credential status information in the iOS keychain.

[Android] The Boxer application implements the functionality to securely store S/MIME and server certificate revocation status information in the encrypted Boxer database.

8.1.16 **[APP_PP] FCS_TLSC_EXT.1(1) & [APP_PP] FCS_TLSC_EXT.1(2):**

The TOE invokes the platform to establish the TLS v1.2 communications channel. TLS v1.2 is the only version supported.

When communicating with the Exchange server, the platform is configured to use the following ciphersuites:

[Android]
TLS_RSA_WITH_AES_128_CBC_SHA as defined in RFC 5246,
TLS_ECDHE_ECDSA_WITH_AES_128_GCM_SHA256 as defined in RFC 5289,
TLS_ECDHE_ECDSA_WITH_AES_256_GCM_SHA384 as defined in RFC 5289,
TLS_ECDHE_RSA_WITH_AES_128_GCM_SHA256 as defined in RFC 5289,
TLS_ECDHE_RSA_WITH_AES_256_GCM_SHA384 as defined in RFC 5289

[iOS]
TLS_RSA_WITH_AES_128_CBC_SHA as defined in RFC 5246,
TLS_RSA_WITH_AES_256_CBC_SHA256 as defined in RFC 5246,
TLS_ECDHE_ECDSA_WITH_AES_128_GCM_SHA256 as defined in RFC 5289,
TLS_ECDHE_ECDSA_WITH_AES_256_GCM_SHA384 as defined in RFC 5289,
TLS_ECDHE_RSA_WITH_AES_128_GCM_SHA256 as defined in RFC 5289,
TLS_ECDHE_RSA_WITH_AES_256_GCM_SHA384 as defined in RFC 5289

The TOE (Android & iOS) verifies the reference identifier during the key exchange process per RFC 6125. The DNS Name and IP address are the only reference identifier types that are supported and is configurable via UEM by an UEM administrator.

The TLS channel will only be established if the peer certificate has been validated. This validation includes checks for the validity of the Common Name (CN) or, if present, the Subject Alternative Name (SAN) along with any extendedKeyUsage field information. DNS Name and IP addresses are the only supported SAN types. The CN and any SAN information is checked against the reference identifier. Wildcards can be used only if it is located in the leftmost portion of the resource identifier. Certificate pinning is not supported.

[iOS] The specific TLS implementation details for iOS can be found in the Apple Security Target documentation referenced in section 1.1.5 of this document.

[Android] The specific TLS implementation details for Android can be found in the Samsung Security Target documentation referenced in section 1.1.5 of this document.

8.1.17 **[APP_PP] FCS_TLSC_EXT.2:**

The application supports mutual TLS authentication using client-side X.509v3 certificates for TLS connections to the Exchange server.

8.1.18 **[APP_PP] FCS_TLSC_EXT.4:**

The supported Elliptic Curves Extension presented in the Client Hello include NIST curves secp256r1 and secp384r1. The NIST curves are available by default.

8.2 User Data Protection

8.2.1 **[APP_PP] FDP_DAR_EXT.1:**

The TOE relies on the underlying platforms (iOS and Android) to provide data-at-rest encryption for all saved data files. The TOE platform must be configured to enforce the use of full device encryption for both iOS and Android providing the first layer of encrypted protection.

The entire internal database is encrypted as well. The TOE’s internal database uses SQLCipher AES-256-CBC provided by OpenSSL (for Android) and iOS encryption library for iOS. The storage of an encrypted database on a fully encrypting the device platform provides a second layer of encrypted protection.

A third layer of protection is provided for all certificate keys. The certificate keys are encrypted prior to storing into the internal encrypted database.

[Android] 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 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. Sensitive data includes: calendar, address book (i.e. contacts), system accounts, profile are stored in an internal database that is fully encrypted in addition to the full disk encryption (double encryption).

[iOS] The 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. Sensitive data includes: calendar, address book, system accounts, profile are stored in an internal database that is fully encrypted in addition to the full disk encryption (double encryption). All instances where files are stored rely on the NSFileProtectionCompleteUntilFirstUserAuthentication declaration in "VMwareBoxer.entitlements", which is contained in the TOE .ipa file. It is enforced in code, except where another Protection class is explicitly used, if any.

8.2.2 [APP_PP] FDP_DEC_EXT.1(1) & [APP_PP] FDP_DEC_EXT.1(2):

The TOE restricts its access and provides user awareness for the intent to access the following hardware resources:

[Android] The hardware access is defined as part of the manifest that can be accessed from the Play Store. It will also be displayed on-access.

- network connectivity
- camera
- NFC
- device storage
- phone
- fingerprint

Access to the following sensitive information repositories are also defined in the manifest and will be displayed on-access.

- address book (i.e. contacts)
- calendar

[iOS] The iOS displays the warnings when access is required.

- network connectivity
- camera
- device storage
- phone
- touch/face ID

The product also displays warnings before accessing the following sensitive information:

- address book (i.e. contacts)
- calendar

8.2.3 [APP_PP] FDP_NET_EXT.1:

The TOE requires network access to facilitate remote administration as well as communicating with the Exchange server. The TOE supports the following user-initiated communications: Send email, force email sync with server, email search, and Global Address List (GAL) lookup. The TOE will respond to communications requests for: ActiveSync (calendar, address book, email). All of these connections are protected using TLS.

[Android] Additionally, Boxer (the TOE) has a “com.google.gms.google-services” dependency, part of Google Services. This is used by Firebase Cloud Message service, which in turn is used for Exchange Notification Service (ENS). The request made to 35.190.25.25:443 is the service initiating a connection to the Google Cloud. That address (based on an ARIN lookup) belongs to Google Cloud. The request is sent out even if ENS is not registered from within Boxer.

[iOS] Additionally, Boxer (the TOE) initializes the following connections besides connections to UEM (e.g. 12.70.190.205, 12.70.190.206) for remote management, and Exchange (e.g. 12.70.190.204) for email functionality:

- Amazon AWS (e.g. ec2-54-193-253-2.us-west-1.compute.amazonaws.com) observed are from Apteligent which Boxer uses for reporting crashes and exceptions.
- Google Cloud Services (e.g. 159.240.178.107.bc.googleusercontent.com) is a Mixpanel collection endpoint. Boxer uses Mixpanel for analytics.
- IP address 192.0.73.2 is for Gravatar. Gravatar services is used by Boxer to fetch email profile images for a sender of an email using the sender's email address.

8.2.4 **[EC_EP] FDP_NOT_EXT.1:**

When the email content is viewed, the TOE shows a notification icon between the header and the body of the email. A seal symbol indicates the email has been signed. A lock symbol indicates the email has been encrypted. Both symbols are displayed when the email has been signed and encrypted. The notification icon uses color coding to help the user quickly identify if there are any issues with the validity of the signer or email. The icon is: black to indicate the certificate is verified and trusted; orange to indicate the email is from an untrusted signer; red to indicate the email has been tampered with. If the email is from an untrusted signer then the user is provided with option to manually accept the certificate.

When displaying the list of emails, a seal symbol is used to notify the user that the email is signed and a lock symbol is used to notify the user that the email is encrypted. Both symbols are displayed if the mail is signed and encrypted. Signature validity is not displayed until the message is opened.

8.2.5 **[EC_EP] FDP_SMIME_EXT.1:**

The TOE uses S/MIME for signing, encrypting, verifying, and decrypting email. S/MIME is implemented as specified in FCS_SMIME_EXT.1. The signature verification and decryption occur at the receipt of the message. The messages are not stored with their S/MIME envelopes.

8.3 Identification and Authentication

8.3.1 **[APP_PP] FIA_X509_EXT.1:**

The TOE application, regardless of platform, performs certificate validation for certificates used for TLS communications. The following is checked in order to determine if a given certificate is valid:

- Certificate validation and certificate path validation conforms to RFC 5280.
- The certificate path must terminate with a trusted CA certificate.
- All CA certificates must have the basicConstraints extension present and the CA flag set to TRUE.
- The TOE uses the Online Certificate Status Protocol (OCSP) as specified in RFC 2560 to verify revocation status. The certificate must not be revoked.
- The extendedKeyUsage field must be valid based on the following rules:
  - Certificates used for trusted updates and executable code integrity verification must have the Code Signing purpose (id-kp 3 with OID 1.3.6.1.5.5.7.3.3) in the extendedKeyUsage field.

  - Server certificates presented for TLS must have the Server Authentication purpose (id-kp with OID 1.3.6.1.5.5.7.3.1) in the extendedKeyUsage field.
  - Client certificates presented for TLS must have the Client Authentication purpose (id-kp 2 with OID 1.3.6.1.5.5.7.3.2) in the extendedKeyUsage field.
  - S/MIME certificates presented for email encryption and signature must have the Email Protection purpose (id-kp 4 with OID 1.3.6.1.5.5.7.3.4) in the extendedKeyUsage field.
  - OCSP certificates presented for OCSP responses must have the OCSP signing purpose (id-kp 9 with OID 1.3.6.1.5.5.7.3.9) in the extendedKeyUsage field.
  - Server certificates presented for EST must have the CMC Registration Authority (RA) purpose (id-kp-cmcRA with OID 1.3.6.1.5.5.7.3.28) in the extendedKeyUsage field.

The certificate validation service will ensure that all certificate paths terminate with a trusted root CA certificate and that all CA certificates include the basicConstraints extension with the CA flag set to TRUE. Certificate status is validated using OCSP. The certificate validation service will also ensure that the extendedKeyUsage field is properly set for all certificates depending on their intended usage.

8.3.2 **[APP_PP] FIA_X509_EXT.2:**

The TOE uses X.509v3 certificates for TLS authentication to the Exchange server. The use of certificates is enabled by default. However, an administrator may configure the behavior of this function by specifying whether mutual authentication is supported. 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 TLS implementation will automatically reject a certificate if it is found to be invalid. 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 (strict setting).
- Moderate Setting: Accept the certificate if the last revocation status is valid. Reject the certificate if the last known revocation status is unknown or was revoked (moderate setting).

8.3.3 **[EC_EP] FIA_X509_EXT.3:**

X.509v3 certificates as defined by RFC 5280, are used for S/MIME functionality and are transmitted from the UEM server to the Boxer application upon initial launch of the application and subsequent launches if new certificates are available.

When the underlying application/platform cannot reach the OCSP responder, the UEM administrator is able to specify the default action so that the TOE will either:

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

The email client will prevent the encryption and/or signing of an email if the email protection certificate is deemed invalid.

8.4 Security Management

8.4.1 [APP_PP] FMT_CFG_EXT.1:

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. The TOE software is installed by default with the appropriate permissions to prevent unauthorized access. Both the Android and the iOS platforms are forced to provide full disk encryption as part of the required environment providing further protection.

8.4.2 [APP_PP] FMT_MEC_EXT.1:

The iOS and Android platforms are configured as UEM assets as part of the installation process. The UEM policy will force the platforms to use full disk encryption to protect the application and its data from unauthorized access. The user of the mobile device must enter a passcode to derive the key which is used for the full disk encryption as part of the process. Secondly, when enrolling the device, it must be enrolled against a server that has a valid pre-existing credentials for the user of the mobile device. Thirdly, the user must enter a passphrase to gain access to the Boxer application and its data. This passphrase is used to derive the Boxer DB key that is required to decrypt the Boxer DB contents for use when needed. The keys and passcodes are stored as described in FCS_STO_EXT.1(1).

[Android] The only user configurable sensitive TSF datum is the application password as defined in FMT_MOD_EXT.1, is stored in the shared preferences XML file as a hashed string.

[iOS] The user configurable sensitive TSF datum—the application password as defined in FMT_MOD_EXT.1, is stored in the iOS keychain.

8.4.3 [EC_EP] FMT_MOF_EXT.1:

The administrator in this evaluation is considered to be the UEM administrator. The UEM administrator is responsible for setting the configuration of the device and TOE applications using the UEM console that resides in the operational environment.

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, OCSP retrieval frequency, S/MIME cryptographic assignments, and enabling/disabling the plaintext only mode globally. The mobile device user is not capable of changing these settings from the mobile device.

The user is the subject that performs management functions on the TOE itself. The TOE allows the user the ability to change one’s own password/passphrase. See Supplemental Administrative Guidance for Common Criteria document on how the user defined password authorization factor can be changed.

8.4.4 [APP_PP] FMT_SMF.1:

At the TOE application, the User is considered the owner or user of the mobile device for which the TOE is installed. The TOE software provides one function that is considered administrative functionality to the end user according to the [EC_EP] FMT_MOF_EXT.1: change password/passphrase authentication credential. The User of both Android and iOS platforms may change their password/passphrase authentication credentials. For both Android and iOS platforms an UEM administrator can force a password change by making the password complexity stricter (i.e. increase password length). The user would be forced to change their password after having successfully authenticated with their old now non-compliant password.

8.5 Privacy

8.5.1 [APP_PP] FPR_ANO_EXT.1:

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. Free text in an email is outside the TOE's scope of control.

8.6 Protection of the TSF

8.6.1 [APP_PP] FPT_AEX_EXT.1:

The TOE implements several mechanisms to protect against exploitation.

[iOS] The TOE is compiled using the LD_NO_PIE=NO compilation flag to ensure it is a Position Independent Executable (ASLR). All 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. Additionally, mprotect is never invoked with the PROT_EXEC permission. The platform forces applications to write all data within the application working directory (sandbox).

[Android] The TOE is compiled using the -fPIE and -pie compilation flags to ensure it is a Position Independent Executable (ASLR). Memory map (mmap) is never invoked with both the PROT_WRITE and PROT_EXEC permissions. Additionally, mprotect is never invoked. User modifiable files are written to /data/data/com.boxer.email and there are also no executable files.

Additionally, the TOE was compiled using the -fstack-protector-all compilation flag for both platforms.

8.6.2 [EC_EP] FPT_AON_EXT.1:

The TOE does not support the installation of trusted or untrusted add-ons.

8.6.3 [APP_PP] FPT_API_EXT.1:

When installed on a mobile device with the Android OS, the TOE uses only the following supported platform APIs in order to function.

- com.android.databinding:adapters
- com.android.databinding:compiler
- com.android.support.constraint:constraint-layout
- com.android.support:animated-vector-drawable
- com.android.support:appcompat
- com.android.support:appcompat-v7
- com.android.support:cardview
- com.android.support:cardview-v7
- com.android.support:customtabs
- com.android.support:design
- com.android.support:gridlayout
- com.android.support:multidex
- com.android.support:palette
- com.android.support:preference-v14
- com.android.support:preference-v7
- com.android.support:recyclerview
- com.android.support:recyclerview-v7

- com.android.support:support-annotations
- com.android.support:support-compat
- com.android.support:support-core-ui
- com.android.support:support-core-utils
- com.android.support:support-fragment
- com.android.support:support-media-compat
- com.android.support:support-v13
- com.android.support:support-v4
- com.android.support:support-vector-drawable
- com.android.volley:volley:1.1.1

When installed on a mobile device with the iOS, the TOE uses only the following supported platform APIs in order to function.

- Accelerate.framework
- Accounts.framework
- AddressBook.framework
- AdSupport.framework
- AssetsLibrary.framework
- AudioToolbox.framework
- AVFoundation.framework
- CallKit.framework
- CFNetwork.framework
- Contacts.framework
- CoreData.framework
- CoreFoundation.framework
- CoreGraphics.framework
- CoreLocation.framework
- CoreMedia.framework
- CoreMotion.framework
- CoreTelephony.framework
- CoreText.framework
- CoreVideo.framework
- EventKit.framework
- Foundation.framework
- ImageIO.framework
- libiconv
- libresolv
- libsqlite3
- libstdc++
- libxml2
- libz.tbd
- LocalAuthentication.framework
- MapKit.framework
- MediaPlayer.framework
- MessageUI.framework
- MobileCoreServices.framework
- Photos.framework
- QuartzCore.framework
- QuickLook.framework
- SafariServices.framework
- Security.framework
- Social.framework
- SystemConfiguration.framework
- UIKit.framework
- UserNotifications.framework
- WebKit.framework

8.6.4 [APP_PP] FPT_LIB_EXT.1:

The TOE is packaged with several third-party open source libraries in order to function.

When installed on a mobile device with the Android OS, the TOE uses only the following third-party dynamic libraries in order to function.

- libEX_Engine8.so
- libSimdEngine.so
- libc++_shared.so
- libcrypto.1.0.2.so
- libcrypto.so
- libpolarisoffice8.so
- libpolarisofficeSDK.so
- libsqlcipher.so
- libssl.1.0.2.so

When installed on a mobile device with iOS, the TOE uses only the following third-party libraries in order to function.

- AFNetworking.framework/
- Alamofire.framework/
- AppAuth.framework/
- CocoaLumberjack.framework/
- CryptBridge.framework/
- evernote_cloud_sdk_ios.framework/
- FMDB.framework/
- FXForms.framework/
- GRMustache.framework/
- GTMAppAuth.framework/
- GTMOAuth2.framework/
- GTMSessionFetcher.framework/
- HTTPStatusCodes.framework/
- JLRoutes.framework/
- JRSwizzle.framework/
- KissXML.framework/
- libical.framework/
- libPhoneNumber_iOS.framework/
- libswiftAVFoundation.dylib
- libswiftContacts.dylib
- libswiftCore.dylib
- libswiftCoreAudio.dylib
- libswiftCoreData.dylib
- libswiftCoreFoundation.dylib
- libswiftCoreGraphics.dylib
- libswiftCoreImage.dylib
- libswiftCoreLocation.dylib
- libswiftCoreMedia.dylib
- libswiftDarwin.dylib
- libswiftDispatch.dylib
- libswiftFoundation.dylib
- libswiftMetal.dylib
- libswiftObjectiveC.dylib
- libswiftos.dylib
- libswiftQuartzCore.dylib
- libswiftsimd.dylib
- libswiftUIKit.dylib
- MBProgressHUD.framework/
- Mixpanel.framework/
- sasl.framework/
- Shimmer.framework/
- SQLCipher.framework/
- SQLite.framework/
- SwiftyJSON.framework/
- SwiftyTraverson.framework/
- TrustKit.framework/
- URITemplate.framework/
- uservoice_iphone_sdk.framework/
- XLActionController.framework/
- ZipArchive.framework/

8.6.5 [APP_PP] FPT_TUD_EXT.1:

The TOE provides a user with the ability to check the version of the TOE that is currently running on the machine.

> [Android] Within the application there is an About command which will display the version. The Android OS also provides the versioning information by using the App manager. For Android, the version of the TOE is accessible by going to: Settings → About.
>
> [iOS] For the iOS platform the version of the TOE is accessible by going to: Settings → About.

The TOE does not automatically update its own binaries or executable files. The binary code is only modified or replaced if the user manually initiates the update via the platform provided update mechanism.

The application for Android is packaged in .apk format and for iOS it is packaged in .ipa format. The user is able to check for updates to the application by navigating to the Google Play Store (Android) or the Apple Store (iOS).

Prior to the initial 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. All communication between the UEM agent and the UEM server is protected using HTTPS/TLS. This secure channel is considered to be out of the scope of the evaluation.

Updates to the TOE are provided by the Google Play Store (Android) or Apple 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 Store. The Google Play Store/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. Secure communication between the mobile device and the stores is handled by the underlying platform. This secure channel is considered part of the operating environment and is out of the scope of the evaluation.

The TOE relies on the platform OS to uninstall the application and remove all remnants of the software from the device. When the TOE application is uninstalled, the entire package is removed. There are no exceptions or customizations for the uninstall function to leave any traces of files or settings.

8.6.5.1 *Timely 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.

8.7 Trusted Path/Channels

8.7.1 **[APP_PP] FTP_DIT_EXT.1(1) & [APP_PP] FTP_DIT_EXT.1(2):**

The TOE communicates with an Exchange server using ActiveSync for its primary function as an email client. The TOE implements the actual ActiveSync protocol layer communications after it invokes the OS to establish the trusted channel that ActiveSync requires.

[Android] The TOE invokes the platform to provide the TLSv1.2 channel to secure all transmitted data via this external interface. In this instance, the TOE/platform acts as the TLS client to initiate the secure communications to the Exchange server.

[iOS] The TOE invokes the platform to provide the TLSv1.2 channel to secure all sensitive data in transit via this external interface. In this instance, the TOE/platform acts as the TLS client to initiate the secure communication to the Exchange server.

8.7.2 **[EC_EP] FTP_ITC_EXT.1:**

As described above in FTP_DIT_EXT.1, the TOE uses ActiveSync to communicate with the Exchange server to send and receive emails. ActiveSync utilizes TLS v1.2 as discussed in FCS_TLSC_EXT.1. The TOE is able to send and receive data from the operational environment over this channel.