VMware Workspace ONE Unified Endpoint Management Version 1907

Security Target

ST Version: 1.0
February 21, 2020

VMware, Inc.
3401 Hillview Ave.
Palo Alto, CA 94304

Prepared By:

Booz | Allen | Hamilton

delivering results that endure

Cyber Assurance Testing Laboratory
1100 West Street
Laurel, MD 20707

Table of Contents

1 Security Target Introduction 7
1.1 ST Reference 7
1.1.1 ST Identification 7
1.1.2 Document Organization 7
1.1.3 Terminology 7
1.1.4 Acronyms 8
1.1.5 Reference 9
1.2 TOE Reference 10
1.3 TOE Overview 10
1.4 TOE Type 12
2 TOE Description 13
2.1 Evaluated Components of the TOE 13
2.2 Components and Applications in the Operational Environment 13
2.3 Excluded from the TOE 14
2.3.1 Not Installed 14
2.3.2 Installed but Requires a Separate License 14
2.3.3 Installed But Not Part of the TSF 14
2.4 Physical Boundary 15
2.4.1 Hardware 15
2.4.2 Software 15
2.5 Logical Boundary 15
2.5.1 Security Audit 15
2.5.2 Communication 16
2.5.3 Cryptographic Support 16
2.5.4 Identification and Authentication 17
2.5.5 Security Management 17
2.5.6 Protection of the TSF 17
2.5.7 TOE Access 18
2.5.8 Trusted Path/Channels 18
3 Conformance Claims 19

3.1 CC Version....19
3.2 CC Part 2 Conformance Claims....19
3.3 CC Part 3 Conformance Claims....19
3.4 PP Claims....19
3.5 Package Claims....19
3.6 Package Name Conformant or Package Name Augmented....20
3.7 Conformance Claim Rationale....20
3.8 Technical Decisions....21
4 Security Problem Definition....22
4.1 Threats....22
4.2 Organizational Security Policies....22
4.3 Assumptions....22
4.4 Security Objectives....23
4.4.1 TOE Security Objectives....23
4.4.2 Security Objectives for the Operational Environment....24
4.5 Security Problem Definition Rationale....24
5 Extended Components Definition....25
5.1 Extended Security Functional Requirements....25
5.2 Extended Security Assurance Requirements....25
6 Security Functional Requirements....26
6.1 Conventions....26
6.2 Security Functional Requirements Summary....26
6.3 Security Functional Requirements....28
6.3.1 Class FAU: Security Audit....28
6.3.2 Class FCO: Communication....34
6.3.3 Class FCS: Cryptographic Support....34
6.3.4 Class FIA: Identification and Authentication....36
6.3.5 Class FMT: Security Management....39
6.3.6 Class FPT: Protection of the TSF....46
6.3.7 Class FTA: TOE Access....47
6.3.8 Class FTP: Trusted Path/Channels....47

6.4 Statement of Security Functional Requirements Consistency 49
7 Security Assurance Requirements 50
7.1 Class ASE: Security Target evaluation 50
7.1.1 ST introduction (ASE_INT.1) 50
7.1.2 Conformance claims (ASE_CCL.1) 51
7.1.3 Security objectives for the operational environment (ASE_OBJ.1) 52
7.1.4 Extended components definition (ASE_ECD.1) 52
7.1.5 Stated security requirements (ASE_REQ.1) 53
7.1.6 TOE summary specification (ASE_TSS.1) 54
7.2 Class ADV: Development 55
7.2.1 Basic Functional Specification (ADV_FSP.1) 55
7.3 Class AGD: Guidance Documentation 56
7.3.1 Operational User Guidance (AGD_OPE.1) 56
7.3.2 Preparative Procedures (AGD_PRE.1) 57
7.4 Class ALC: Life Cycle Support 57
7.4.1 Labeling of the TOE (ALC_CMC.1) 57
7.4.2 TOE CM Coverage (ALC_CMS.1) 58
7.5 Class ATE: Tests 58
7.5.1 Independent Testing - Conformance (ATE_IND.1) 58
7.6 Class AVA: Vulnerability Assessment 59
7.6.1 Vulnerability Survey (AVA_VAN.1) 59
8 TOE Summary Specification 60
8.1 Security Audit 61
8.1.1 [MDMPP] FAU_ALT_EXT.1 61
8.1.2 [AGENTMOD] FAU_ALT_EXT.2/ANDROID 62
8.1.3 [AGENTMOD] FAU_ALT_EXT.2/IOS 63
8.1.4 [MDMPP] FAU_GEN.1(1) 64
8.1.5 [MDMPP] FAU_GEN.1(2) 67
8.1.6 [AGENTMOD] FAU_GEN.1(2) 67
8.1.7 [MDMPP] FAU_NET_EXT.1 69
8.1.8 [MDMPP] FAU_SAR.1 69

8.1.9 [AGENTMOD] FAU_SEL.1(2) 70
8.1.10 [MDMPP] FAU_STG_EXT.1 70
8.2 Communication 71
8.2.1 [MDMPP] FCO_CPC_EXT.1 71
8.3 Cryptographic Support 71
8.3.1 [MDMPP] FCS_CKM.1 71
8.3.2 [MDMPP] FCS_CKM.2 72
8.3.3 [MDMPP] FCS_CKM_EXT.4 72
8.3.4 [MDMPP] FCS_COP.1(1) 73
8.3.5 [MDMPP] FCS_COP.1(2) 73
8.3.6 [MDMPP] FCS_COP.1(3) 73
8.3.7 [MDMPP] FCS_COP.1(4) 74
8.3.8 [MDMPP] FCS_RBG_EXT.1 74
8.3.9 [MDMPP] FCS_STG_EXT.1 74
8.3.10 [AGENTMOD] FCS_STG_EXT.1(2) 75
8.4 Identification and Authentication 76
8.4.1 [MDMPP] FIA_ENR_EXT.1/ANDROID 76
8.4.2 [MDMPP] FIA_ENR_EXT.1/IOS 76
8.4.3 [AGENTMOD] FIA_ENR_EXT.2 77
8.4.4 [MDMPP] FIA_UAU.1 77
8.4.5 [MDMPP] FIA_X509_EXT.1(1) and [MDMPP] FIA_X509_EXT.2 77
8.4.6 [MDMPP] FIA_X509_EXT.5 80
8.5 Security Management 80
8.5.1 [MDMPP] FMT_MOF.1(1) 80
8.5.2 [MDMPP] FMT_MOF.1(2) 81
8.5.3 [MDMPP] FMT_MOF.1(3) 81
8.5.4 [MDMPP] FMT_POL_EXT.1 82
8.5.5 [AGENTMOD] FMT_POL_EXT.2 82
8.5.6 [MDMPP] FMT_SMF.1(1)/ANDROID and [MDMPP] FMT_SMF.1(1)/IOS 82
8.5.7 [MDMPP] FMT_SMF.1(2)/ANDROID and [MDMPP] FMT_SMF.1(2)/IOS 88
8.5.8 [MDMPP] FMT_SMF.1(3) 89

8.5.9 [AGENTMOD] FMT_SMF_EXT.4 89
8.5.10 [MDMPP] FMT_SMR.1(1) 90
8.5.11 [MDMPP] FMT_SMR.1(2) 91
8.5.12 [AGENTMOD] FMT_UNR_EXT.1 91
8.6 Protection of the TSF 91
8.6.1 [MDMPP] FPT_API_EXT.1 91
8.6.2 [MDMPP] FPT_ITT.1(2) 92
8.6.3 [MDMPP] FPT_LIB_EXT.1 92
8.6.4 [MDMPP] FPT_TST_EXT.1 93
8.6.5 [MDMPP] FPT_TUD_EXT.1 93
8.7 TOE Access 93
8.7.1 [MDMPP] FTA_TAB.1 93
8.8 Trusted Path/Channels 94
8.8.1 [MDMPP] FTP_ITC_EXT.1 94
8.8.2 [MDMPP] FTP_ITC.1(1) 94
8.8.3 [MDMPP] FTP_TRP.1(1) 94
8.8.4 [MDMPP] FTP_TRP.1(2) 95
9 Appendix A: List of Third-Party Libraries 96
9.1 Windows Server 2016 Libraries 96
9.2 Android 9 Libraries 4
9.3 iOS 12 Libraries 1

Table of Figures

Figure 1: TOE Boundary 11

Table of Tables

Table 1: Customer-Specific Terminology 7
Table 2: CC-Specific Terminology 8
Table 3: Acronym Definitions 8

Table 4: Evaluated Components of the TOE 13
Table 5: Components of the Operational Environment 13
Table 6: Cryptographic Algorithm Table for the Hub Agents 16
Table 7: TOE Threats 22
Table 8: TOE Organizational Security Policies 22
Table 9: TOE Assumptions 23
Table 10: TOE Objectives 23
Table 11: Operational Environment Objectives 24
Table 12: Security Functional Requirements for the TOE 26
Table 13: Server Auditable Events 30
Table 14: Agent Auditable Events 32
Table 15: SFR and TOE Component Mapping 60
Table 16: Auditable Events by Enforcing Component 64
Table 17: Agent Auditable Events by Enforcing Component 68
Table 18: Keys and CSPs for the UEM Server Platform 74
Table 19: Keys and CSPs for the Device 75
Table 20: UEM Server Management Functions 83

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 Unified Endpoint Management Version 1907 Security Target
ST Version:	1.0
ST Publication Date:	February 21, 2020
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.

Table 1: Customer-Specific Terminology

Term	Definition
End User	An individual who possesses a mobile device that is managed by VMware and who has limited authority to perform management functions using the Self-Service Portal

Role	The level of access given to Administrator accounts. The TOE comes with pre-defined roles but new roles with custom sets of privileges can be created.
System Administrator	The class of TOE Administrators that have complete access to a VMware environment, including the underlying Windows Server 2016 platform.

Table 2: CC-Specific Terminology

Term	Definition
Administrator	The claimed Protection Profile defines an Administrator as the person who is responsible for management activities, including setting the policy that is applied by the enterprise on the mobile device. This TOE defines separate user roles.
Authorized Administrator	Synonymous with Administrator.
MD User	User with a mobile device (MD).
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.

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.

Table 3: Acronym Definitions

Acronym	Definition
APNS	Apple Push Notification Service
CA	Certificate Authority
CC	Common Criteria
CPU	Central Processing Unit
CSP	Critical Security Parameter
DEP	[Apple] Device Enrollment Program
FCM	[Android] Firebase Cloud Messaging [Service]
FQDN	Fully Qualified Domain Name
GUI	Graphical User Interface
HTTP	Hypertext Transfer Protocol
HTTPS	Hypertext Transfer Protocol Secure over a bidirectional TLS encrypted tunnel
IMEI	International Mobile Equipment Identity
IP	Internet Protocol
IIS	Internet Information Services
IT	Information Technology
LDAP	Lightweight Directory Access Protocol
MAS	Mobile Application Store
MD	Mobile Device
MDM	Mobile Device Management
NFC	Near-Field Communication
NIAP	National Information Assurance Partnership
OS	Operating System
OSP	Organizational Security Policy

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
UI	User Interface
USB	Universal Serial Bus
VPN	Virtual Private Network
WLAN	Wireless Local Area Network

1.1.5 Reference

[1] Protection Profile for Mobile Device Management, version 4.0 [MDMPP]

[2] Protection Profile Module for Mobile Device Management Agents, version 1.0 [AGENTMOD]

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

[4] Common Criteria for Information Technology Security Evaluation – Part 2: Security functional components, dated April 2017, version 3.1, Revision 5, CCMB-2017-04-002

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

[6] Common Methodology for Information Technology Security Evaluation – Evaluation Methodology, dated April 2017, version 3.1, Revision 5, CCMB-2017-04-004

[7] NIST Special Publication 800-56A Revision 3: Recommendation for Pair-Wise Key Establishment Schemes Using Discrete Logarithm Cryptography, April 2018

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

[9] NIST Special Publication 800-38D Recommendation for Block Cipher Modes of Operation: Galois/Counter Mode (GCM) and GMAC, November 2007

[10] NIST Special Publication 800-90A Revision 1, Recommendation for Random Number Generation Using Deterministic Random Bit Generators, June 2015

[11] NIST Special Publication 800-57 Part 1 Revision 4, Recommendation for Key Management, January 2016

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

[13] FIPS PUB 180-4 Federal Information Processing Standards Publication Secure Hash Standard (SHS), August 2015

[14] FIPS PUB 186-4 Digital Signature Standard (DSS), July 2013

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

[16] FIPS PUB 198-1 Federal Information Processing Standards Publication The Keyed-Hash Message Authentication Code (HMAC), July 2008
[17] VMware Workspace ONE Unified Endpoint Management Supplemental Administrative Guidance v1.0
[18] Apple iPad and iPhone Mobile Devices with iOS 12 Security Target, Version 2.0 (VID 10937)
[19] Samsung Electronics Co., Ltd. Samsung Galaxy Devices on Android 9 (MDFPP31/WLANCEP10/VPNC21) Security Target, Version 0.5 (VID 10979)
[20] Microsoft Windows Common Criteria Evaluation Microsoft Windows 10 version 1803 (April 2018 Update) Microsoft Windows Server version 1803 (April 2018 Update) Security Target, Version 0.04

1.2 TOE Reference

The TOE is the VMware Workspace ONE Unified Endpoint Management version 1907 comprising of the Unified Endpoint Management Server and one or more VMware Intelligent Hub Agents installed on Apple and Android devices. The minimum configuration for this evaluation is one Unified Endpoint Management Server, and one VMware Intelligent Hub Agent installed on an Apple device and/or one VMware Intelligent Hub Agent installed on an Android device. Including additional VMware Intelligent Hub Agents installed on multiple Apple devices and additional VMware Intelligent Hub Agents installed on multiple Android devices as part of an operational configuration will not affect the validity of the functional claims made within this document and the Common Criteria certification.

1.3 TOE Overview

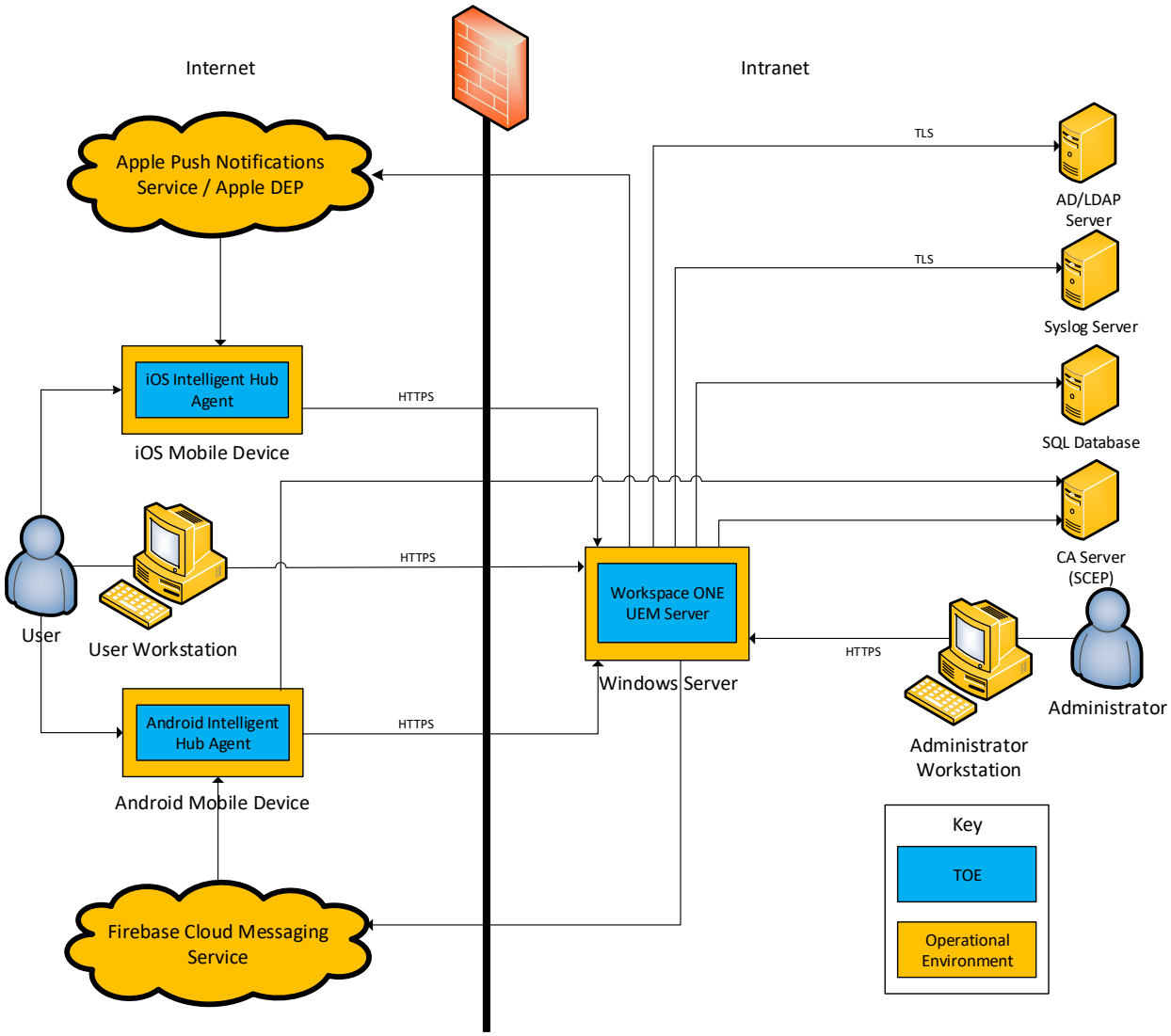
The TOE is a Mobile Device Management product and is comprised of an MDM Server component (UEM Server) and one or more VMware Intelligent Hub Agent components (iOS Hub Agent and Android Hub Agent). In the evaluated configuration of the TOE, the UEM Server is deployed in an on-premises configuration. The UEM Server component provides a centralized enterprise level management capability for a collection of mobile devices running the iOS and Android Hub Agents. The UEM Server is also a Mobile Application Store (MAS) Server that allows managed devices to download apps from a trusted repository that resides within the organization managing the device. The management functionality includes management of Administrators and users, mobile device enrollment, mobile device status, mobile device compliance and policy management, and application management. Administrators access the UEM Server through the Admin Console interface in order to manage users, policies, and devices. Users access the UEM Server through the Self-Service Portal, which allows them to perform administrative functions relating to their own devices.

The UEM Server runs on a Microsoft Windows Server 2016 operating system and authentication to the UEM Server is provided by the Active Directory/LDAP service. The UEM Server also provides local authentication for initial setup and to mitigate a denial of service in the event the Active Directory/LDAP service is unavailable. The UEM Server stores audit data remotely in a SQL database but can send audit records via a TLS encrypted trusted channel to a remote Syslog Server for remote storage.

In the evaluated configuration, the Hub Agent runs on mobile devices with either Apple iOS 12 or Android 9 operating systems. The communication channel between the Hub Agent and the UEM Server is protected

by TLS. Apple DEP is used to enroll the Apple devices with the UEM Server so that it can be managed by the UEM Server. The Hub Agents provide status and policy information about the mobile device to the UEM Server. Figure 1 depicts the network configuration of the TOE.

Figure 1: TOE Boundary



As depicted in Figure 1, the TOE consists of a UEM Server and one or more instances of the iOS and Android Hub Agents running on mobile devices. The expected deployment of the TOE is to have an on-premises deployment of the UEM Server running behind the firewall. The UEM Server hosts the Self-Service Portal so that enterprise users can enroll and manage their own devices from outside the firewall and the Admin Console resides behind the firewall. The connection between the iOS and Android Hub Agent devices and the UEM Server is also protected by HTTPS. The connections between the UEM Server, and the Syslog Server and AD/LDAP Server are protected with TLS. The UEM Server connects to a SQL Database to store configuration information. The UEM Server is also connected to a CA Server in the internal network for the purposes of issuing unique certificates to the Hub Agents.

1.4 TOE Type

The TOE is a Mobile Device Management product consisting of UEM Server software and one or more Hub Agents which runs on mobile devices. The [MDMPP] states:

"The MDM Server is software (an application, service, etc.) on a general-purpose platform, a network device, or cloud architecture executing in a trusted network environment. The MDM Server provides administration of the mobile device policies and reporting on mobile device behavior. The MDM Server is responsible for managing device enrollment, configuring and sending policies to the MDM Agents, collecting reports on device status, and sending commands to the Agents. The MDM Server may be standalone or distributed, where a distributed TOE is one that requires multiple distinct components to operate as a logical whole in order to fulfill the requirements of this PP."

The MDM Server TOE type is justified because the TOE software provides centralized enterprise level management capabilities for MDM Agents (iOS and Android Hub Agents) running on mobile devices, including enrollment, policy management and device status and the MDM Server (UEM) runs on Microsoft Windows Server 2016, which is a general-purpose platform.

The [MDMPP] also states:

"The MDM Agent establishes a secure connection back to the MDM Server controlled by an enterprise administrator and configures the mobile device per the administrator's policies. The MDM Agent is addressed in the Module for MDM Agents. If the MDM Agent is installed on a mobile device as an application developed by the MDM developer, the extends this PP and is included in the TOE. In this case, the TOE security functionality specified in this PP must be addressed by the MDM Agent in addition to the MDM Server. Otherwise, the MDM Agent is provided by the mobile device vendor and is out of scope of this PP; however, MDMs are required to indicate the mobile platforms supported by the MDM Server and must be tested against the native MDM agent of those platforms."

This statement is re-iterated in the [AGENTMOD]. The MDM Agent TOE type is justified because the TOE Agent software (iOS and Android Hub Agents) is installed on a mobile device as an application developed by VMware and establishes a secure connection back to the MDM Server (UEM Server) protected by HTTPS.

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:

Table 4: Evaluated Components of the TOE

Component	Definition
Workspace ONE Unified Endpoint Management 1907 (UEM Server)	This satisfies the MDM Server Component of the TOE as it provides an enterprise-level management capability for a collection of mobile devices, including the administration of mobile device policies, reporting on device behavior, and sending commands to the iOS and Android Hub Agent(s). This MDM Server Component also provides a Mobile Application Store (MAS) Server that allows managed devices to download apps from a trusted repository.
Android Intelligent Hub Agent 19.08 (Android Hub Agent)	This satisfies the MDM Agent Component of the TOE as it is a VMware-developed application installed on mobile devices running the Samsung Android 9 operating system and uses the Android platform to establish a secure connection back to the UEM Server for the Android Hub Agent can provide status and policy information about the device.
iOS Intelligent Hub Agent 19.09 (iOS Hub Agent)	This satisfies the MDM Agent Component of the TOE as it is a VMware-developed application installed on mobile devices running the Apple iOS 12 operating system and uses the iOS platform to establish a secure connection back to the UEM Server for the iOS Hub Agent and iOS platform to provide status and policy information about the device.

As shown in Figure 1, the TOE boundary on the end user mobile devices includes only the iOS and Android Hub Agents itself; the actual devices have been evaluated against the Mobile Device Fundamentals Protection Profile under the Validation ID number identified in Table 5 below.

2.2 Components and Applications in the Operational Environment

The following table lists components and applications in the TOE's operational environment that must be present for the TOE to be operating in its evaluated configuration:

Table 5: Components of the Operational Environment

Component	Definition
Active Directory / LDAP Server	Identity store that defines users for device enrollment and administrator accounts for access to the Admin Console. In the evaluated configuration, Windows Server 2016 (Version 1803) Active Directory/LDAP Server is used.
Apple iOS 12 Mobile Device (VID10937)	The MDM Agent Component of the TOE (Hub Agent) is an application that is installed on Apple mobile devices running iOS 12 operating systems so that the TOE can provide management functionality to the device.
Apple Push Notification Service (APNS) / Apple DEP	APNS is an iOS platform push notification service that enables the UEM Server to notify iOS Hub Agents and the iOS platform to connect directly to the UEM Server to retrieve data (e.g. policies). Apple DEP is an online service that automates the enrollment of iOS devices into the TOE in the evaluated configuration.

Certification Authority (CA) Server	The MDM Server Component and Android Hub Agent of the TOE connect to the CA Server during device enrollment so that the TOE can provide each device with a unique certificate generated by the CA Server. In the evaluated configuration, Windows Server 2016 (Version 1803) Active Directory Certificate Services is used.
Firebase Cloud Messaging Service (FCM)	FCM is an Android platform push notification service that enables the UEM Server to notify Android Hub Agents to connect directly to the UEM Server to retrieve data (e.g. policies).
Samsung Android 9 Mobile Device (VID 10979)	The MDM Agent Component of the TOE (Hub Agent) is an application that is installed on mobile devices running Android 9 operating systems so that the TOE can provide management functionality to the device.
SQL database	The TOE's RDBMS database used to store configuration settings and device data. In the evaluated configuration, Microsoft SQL Server 2012 Enterprise is used.
Syslog Server	The MDM Server Component of the TOE connects to the Syslog Server to persistently store audit data for the UEM Server's own operation as well as the audit data collected from the Hub Agent that it manages.
Windows Server 2016 (Version 1803)	This is the OS that the UEM Server is installed on.
Workstation	Any general-purpose computer that is used by an administrator to manage the TOE via the Admin Console and a user to manage their device via the Self-Service Portal. For the TOE to be accessed remotely, the workstation is required to have a browser to access the TOE's GUI based interfaces.

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 optional components that are omitted from the installation process.

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 satisfies 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.

In particular, note that the VMware product also includes a Secure Email Gateway and Mobile Access Gateway. These components have not been evaluated because their functionality is outside the scope of the

claimed Protection Profiles. However, their presence in the Operational Environment does not interfere with the security enforcement of the TSF and therefore can be deployed in an environment with the TOE.

2.4 Physical Boundary

2.4.1 Hardware

The TOE is a software product. All hardware that is present is part of the TOE's Operational Environment.

2.4.2 Software

The VMware Workspace ONE UEM runs on Windows Server 2016 (Version 1803) and includes the Admin Console, Self-Service Portal, UEM Server, and MAS Server functions. The software version of the VMware Workspace ONE UEM is 1907 (displayed as 19.7).

The VMware Hub Agent runs on Apple iOS 12 and Android 9 operating systems in its evaluated configuration. The software version of the iOS Hub Agent is 19.09. The software version of the Android Hub Agent is 19.08 and includes OpenSSL version 2.0.16.

Refer to the VMware Workspace ONE Unified Endpoint Management Supplemental Administrative Guidance version 1.0 for information on delivery of the TOE software.

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 both UEM Server and the Hub Agent functionality.

1. Security Audit
2. Communication
3. Cryptographic Support
4. Identification and Authentication
5. Security Management
6. Protection of the TSF
7. TOE Access
8. Trusted Path/Channels

2.5.1 Security Audit

The UEM Server component of the TOE creates audit records for auditable events related to administrative actions, configuration of the UEM Server itself, and server-initiated management activities that affect one or more managed mobile devices. The UEM Server's MAS Server functionality also generates audit records when it experiences a failure to push or update an application on a managed mobile device. The audit records are stored in an SQL database and are transferred to a remote Syslog Server over a TLS encrypted trusted channel. Audit records can be viewed on the Admin Console.

The UEM Server can issue 'compliance policies' to managed mobile devices. Compliance policies are used to compare the configuration, status, or characteristics of a mobile device against a certain baseline

and can be used to generate an alert to an Administrator if an anomaly is detected. The Administrator can also request on-demand connectivity status updates through the use of push notifications.

iOS and Android Hub Agents' audit records are created as long as the underlying mobile device is powered on. The iOS and Android Hub Agents generate audit records for the activities it performs as a result of its interactions with the UEM Server or as a result of stored policy information. The iOS and Android Hub Agents facilitate alerts by providing data to the UEM Server on a periodic basis. The UEM Server can then analyze this data (or the absence of data in the case of periodic reachability events) in order to determine if anomalous behavior is occurring.

2.5.2 Communication

The iOS and Android Hub Agents mobile devices are registered with the UEM Server so they can be enrolled into management by the UEM Server. This requires an Administrator to enable communications between these TOE components by including the mobile device's identifier in a whitelist of devices that are allowed to enroll on the UEM Server. The enrollment process occurs over an HTTPS/TLS trusted channel that is handled by each TOE components' underlying platform. An Administrator can disable the communications between an iOS or Android Hub Agent and the UEM Server by performing a wipe of the Hub Agent's mobile device.

2.5.3 Cryptographic Support

The UEM Server invokes the Windows Server 2016 platform for cryptographic services to establish TLS and HTTPS/TLS trusted channels and paths to ensure secure communications of data in transit. This includes the use of RSA and Elliptic Curve Cryptography (ECC) key establishment techniques. The MAS Server is integrated with the UEM Server, so it invokes the same cryptography services. The UEM Server also invokes the Windows Server 2016 platform to digitally sign policies sent to the Hub Agents.

The iOS and Android Hub Agents invoke their underlying mobile device platforms (Apple iOS 12 and Android 9 respectively) for cryptographic services to also establish trusted communications. The iOS Hub Agent invokes its underlying platform to verify the digital signatures of the all policies received from the UEM Server. The Android Hub Agent software contains an OpenSSL library for implementing the digital signature verification of the all policies received from the UEM Server.

All cryptographic mechanisms use the TOE components' platform provided DRBG functionality to support their cryptographic operations. Cryptographic functionality includes encryption/decryption services, credential/key storage, key establishment, key destruction, hashing services, signature services, and hashed message authentication.

The following table contains the CAVP algorithm certificates corresponding to the Android Hub Agent's digital signature verification cryptographic functionality which is implemented by its OpenSSL module.

Table 6: Cryptographic Algorithm Table for the Hub Agents

	Algorithm	CAVP Cert. # (Android 9)
FCS_COP.1(2) – Hashing Algorithms	SHA-512	C1329
FCS_COP.1(3) – Signature Algorithms	ECDSA with P-521 NIST curve	C1329

2.5.4 Identification and Authentication

The iOS and Android Hub Agents register with the UEM Server so that their mobile device can be enrolled into management by the UEM Server. The mobile device user that is performing the enrollment must have a user account on the UEM Server to access the Self-Service Portal and authenticate to the TOE. During the enrollment process, the iOS and Android Hub Agents record the UEM Server's DNS name and full URL with hostname. The iOS and Android Hub Agents also receive a unique certificate during enrollment that is used to establish an HTTPS trusted channel with the UEM Server.

Administrators (through the Admin Console) and users (through the Self-Service Portal) cannot access the UEM Server without being authenticated. Administrators and users can view the configured pre-authentication warning banner and query the UEM Server's software version number prior to authentication.

The UEM Server interfaces with the underlying Windows Server 2016 platform to provide certificate validation services. Certificates are used for HTTPS/TLS authentication, code signing for software updates, code signing for integrity verification, and signing of MDM policies. The iOS and Android Hub Agents rely on the underlying platform to perform all certificate validation services, except for policy signing on Android devices which is validated by the Android Hub Agent's implementation of OpenSSL.

2.5.5 Security Management

The TSF provides separate administrative interfaces for Administrators and for mobile device users. Administrators use the Admin Console to manage users, policies, and devices, while MD users use the Self-Service Portal to perform actions related to their own devices. The mobile device user installs the TOE's iOS or Android Hub Agent on the mobile device which will communicate with the UEM Server to enroll in management. Once enrolled, the TOE will prevent user-directed unenrollment from management.

The UEM Server can be used to transmit specific commands to a managed device such as forcibly locking the device, initiating a wipe operation, or sending a push notification. The UEM Server can also define policies (known as profiles) that specify the configuration settings for a device. These configuration settings can include functionality such as configuration of the password policy and what settings are applied to Wi-Fi connections. The UEM Server transmits iOS policies either to the iOS Hub Agent or iOS platform directly, depending on the functionality being configured. The UEM Server transmits Android policies to the Android Hub Agent. The UEM Server invokes its underlying platform to sign all policy data using ECDSA with SHA-512. The underlying iOS mobile platform and Android Hub Agent will validate the signed policies when they are received.

The UEM Server also includes the MAS Server functionality, which provides the ability to grant or deny access to specific applications stored on the MAS Server to devices or groups of devices. The MAS Server is accessed through the same Admin Console interface as the UEM Server, so the administrative roles defined for both components are the same.

2.5.6 Protection of the TSF

The communications between the UEM Server and iOS and Android Hub Agents are protected using HTTPS/TLS which is provided by the underlying platforms of the TOE components.

The UEM Server invokes its platform to verify the digital signatures of executables and .dlls using Microsoft's Authenticode making use of X.509v3 certificates. In addition, the UEM Server's platform uses FIPS validated cryptographic modules which perform their own integrity checks at startup.

The TOE components invoke their underlying platforms to update their software and the platforms will verify the digital signatures of the updates prior to installing them. The TOE components software contain third party libraries. The TOE components use only documented APIs from their underlying platforms.

2.5.7 TOE Access

The UEM Server displays a pre-authentication banner for the Admin Console and the Self-Service Portal. This can be customized by Administrators to fit the needs of the organization deploying the TOE.

2.5.8 Trusted Path/Channels

The trusted communication channels between the UEM Server and the devices running the iOS and Android Hub Agents, the Syslog Server, and the AD/LDAP Server make use of TLS or HTTPS/TLS, depending on the interface. The trusted communication channels are provided by the TOE components' underlying platforms.

The UEM Server platform uses HTTPS/TLS to provide a trusted path between itself and remote Administrators through the Admin Console and mobile device users through the Self-Service Portal as well as during the enrollment of a mobile device.

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) are Part 2 (extended) to include all applicable NIAP and International interpretations through February 21, 2020.

3.3 CC Part 3 Conformance Claims

This ST and Target of Evaluation (TOE) are Part 3 (conformant) to include all applicable NIAP and International interpretations through February 21, 2020.

3.4 PP Claims

This ST claims exact conformance to the following Protection Profiles:

- Protection Profile for Mobile Device Management, version 4.0 [MDMPP]
- Protection Profile Module for Mobile Device Management Agents, version 1.0 [AGENTMOD]

3.5 Package Claims

The TOE claims exact compliance to the *Protection Profile for Mobile Device Management*, *version 4.0* and *Protection Profile Module for Mobile Device Management Agents*, *version 1.0*, which are conformant with CC Part 3.1.

The TOE claims the following Selection-Based SFRs that are defined in the appendices of the claimed PP:

[MDMPP]:

- FAU_GEN.1(2)
- FMT_MOF.1(3)
- FMT_SMF.1(3)
- FMT_SMR.1(2)
- FPT_ITT.1(2)

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

[MDMPP]:

- FAU_SAR.1
- FTA_TAB.1

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

[MDMPP]:

- FCO_CPC_EXT.1

This does not violate the notion of exact conformance because the PP specifically indicates these as allowable selections, options, and objectives, 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 [MDMPP] and [AGENTMOD].

3.7 Conformance Claim Rationale

The [MDMPP] states the following:

"The MDM Server is software (an application, service, etc.) on a general-purpose platform, a network device, or cloud architecture executing in a trusted network environment. The MDM Server provides administration of the mobile device policies and reporting on mobile device behavior. The MDM Server is responsible for managing device enrollment, configuring and sending policies to the MDM Agents, collecting reports on device status, and sending commands to the Agents. The MDM Server may be standalone or distributed, where a distributed TOE is one that requires multiple distinct components to operate as a logical whole in order to fulfill the requirements of this PP"

The [AGENTMOD] states the following:

"The MDM system consists of two primary components: the MDM Server software and the MDM Agent. This PP-Module specifically addresses the MDM Agent. The MDM Agent establishes a secure connection back to the MDM Server, from which it receives policies to enforce on the mobile device. Optionally, the MDM Agent interacts with the Mobile Application Store (MAS) Server to download and install enterprise-hosted applications.

A compliant MDM Agent is installed on a mobile device as an application (supplied by the developer of the MDM Server software) or is part of the mobile device's OS. This PP-Module builds on either the MDF PP or the MDM PP. A TOE that claims conformance to this PP-Module must also claim conformance to one of those PPs as its Base-PP. A compliant TOE is obligated to implement the functionality required in the Base-PP along with the additional functionality defined in this PP Module in order to mitigate the threats that are defined by this PP-Module.

This PP-Module shall build on the MDF PP if the TOE is a native part of a mobile operating system. The TOE for this PP Module combined with the MDF PP is the mobile device itself plus the MDM Agent. If the MDM Agent is part of the mobile device's OS, the MDM Agent may present multiple interfaces for configuring the mobile device, such as a local interface and a remote interface. Agents conforming to this PP-Module must at least offer an interface with a trusted channel that serves as one piece of an MDM system. Conformant MDM Agents may also offer other interfaces, and the configuration aspects of these additional interfaces are in scope of this PP-Module.

This PP-Module shall build on the MDM Server PP if the TOE is a third-party application that is provided with an MDM Server and installed on a mobile device by the user after acquiring the mobile device. The distributed TOE for this PP-Module combined with the MDM Server PP is the entire MDM environment, which includes both the MDM Server and the MDM Agent. Even though the mobile device itself is not part of the TOE, it is expected to be evaluated against the MDF PP so that its baseline security capabilities can be assumed to be present."

The MDM Server component (UEM Server) of the TOE is designed to provide centralized management capabilities of the MDM Agent components (iOS and Android Hub Agents) of the TOE which reside on mobile devices. The iOS Android Hub Agent communicates with the UEM Server over a secure channel. Therefore, the conformance claim is appropriate.

3.8 Technical Decisions

The following is the list of NIAP Technical Decisions that are applicable to the ST/TOE and a summary of their impact:

TD #	Title	References	Changes			Analysis to this evaluation	
			SFR	AA	Notes	NA	Reason
TD0438	TST and TUD on the MDM Agent	[MDMPP] FPT_TST_EXT.1 and [MDMPP] FPT_TUD_EXT.1.1	X		X		Clarifying distributed TOE components
TD0461	Security Audit for Distributed TOEs	[MDMPP] Section 6.2.2, Bullet 2			X		Clarify audit transfer on distributed TOE
TD0467	OCSP Stapling Added as Selection	[MDMPP] FIA_X509_EXT.1 and [MDMPP] FIA_X509_EXT.2	X	X	X		SFR changed but TOE does not use OCSP stapling
TD0479	FMT_SMF.1(1) Reliance on MDF Evals	[MDMPP] FMT_SMF.1(1)		X	X		TOE claims more functions than evaluation
TD0491	Update to FMT_SMF_EXT.4 Test 2	[AGENTMOD] FMT_SMF_EXT.4		X			Updates wording of Test 2

Note that Technical Decisions were not considered to be applicable if any of the following conditions were true:

- The Technical Decision does not apply to the [MDMPP] or [AGENTMOD].
- The Technical Decision does not apply to the current version of the [MDMPP] or [AGENTMOD].
- The Technical Decision applies to an SFR that was not claimed by the TOE.
- The Technical Decision applies to an SFR selection or assignment that was not chosen for the TOE.
- The Technical Decision only applies to one or more Application Notes in the [MDMPP] or [AGENTMOD] and does not affect the SFRs or how the evaluation of the TOE is conducted.
- The Technical Decision is an affirmation that an existing requirement or Evaluation Activity is correct.
- The Technical Decision was superseded by a more recent Technical Decision.
- The Technical Decision is issued as guidance for future versions of the [MDMPP] or [AGENTMOD].

4 Security Problem Definition

4.1 Threats

Note: Unless otherwise stated Threats, Organizational Security Policies (OSPs), Assumptions and Security Objectives apply to both the Agent and Server.

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

Table 7: TOE Threats

Threat	Threat Definition
T.MALICIOUS_APPS	Malicious or flawed application threats exist because apps loaded onto a mobile device may include malicious or exploitable code. An administrator of the MDM or mobile device user may inadvertently import malicious code, or an attacker may insert malicious code into the TOE, resulting in the compromise of TOE or TOE data.
T.BACKUP	[AGENTMOD] An attacker may try to target backups of data or credentials and exfiltrate data. Since the backup is stored on either a personal computer or end user's backup repository, it's not likely the enterprise would detect compromise.
T.NETWORK_ATTACK	An attacker may masquerade as an MDM Server and attempt to compromise the integrity of the mobile device by sending malicious management commands.
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 MDM Server and other endpoints.
T.PHYSICAL_ACCESS	The mobile device may be lost or stolen, and an unauthorized individual may attempt to access user data. Although these attacks are primarily directed against the mobile device platform, the TOE configures features, which address these threats.

4.2 Organizational Security Policies

This section identifies the organizational security policies which are expected to be implemented by an organization that deploys the TOE. These policies have been taken from the [MDMPP] and [AGENTMOD].

Table 8: TOE Organizational Security Policies

Policy	Policy Definition
P.ACCOUNTABILITY	Personnel operating the TOE shall be accountable for their actions within the TOE.
P.ADMIN	The configuration of the mobile device security functions must adhere to the Enterprise security policy.
P.DEVICE_ENROLL	A mobile device must be enrolled for a specific user by the administrator of the MDM prior to being used in the Enterprise network by the user.
P.NOTIFY	The mobile user must immediately notify the administrator if a mobile device is lost or stolen so that the administrator may apply remediation actions via the MDM system.

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 [MDMPP] and [AGENTMOD].

Table 9: TOE Assumptions

Assumption	Assumption Definition
A.COMPONENTS_RUNNING (applies to distributed TOEs only)	[MDMPP] For distributed TOEs it is assumed that the availability of all TOE components is checked as appropriate to reduce the risk of an undetected attack on (or failure of) one or more TOE components. It is also assumed that in addition to the availability of all components it is also checked as appropriate that the audit functionality is running properly on all TOE components.
A.CONNECTIVITY	The TOE relies on network connectivity to carry out its management activities. The TOE will robustly handle instances when connectivity is unavailable or unreliable.
A.MDM_SERVER_PLATFORM	[MDMPP] The MDM Server relies upon a trustworthy platform and local network from which it provides administrative capabilities. The MDM Server relies on this platform to provide a range of security-related services including reliable timestamps, user and group account management, logon and logout services via a local or network directory service, remote access control, and audit log management services to include offloading of audit logs to other servers. The platform is expected to be configured specifically to provide MDM services, employing features such as a host-based firewall, which limits its network role to providing MDM functionality.
A.MOBILE_DEVICE_PLATFORM	[AGENTMOD] The MDM Agent relies upon mobile platform and hardware evaluated against the MDF PP and assured to provide policy enforcement as well as cryptographic services and data protection. The mobile platform provides trusted updates and software integrity verification of the MDM Agent.
A.PROPER_ADMIN	One or more competent, trusted personnel who are not careless, willfully negligent, or hostile, are assigned and authorized as the TOE Administrators, and do so using and abiding by guidance documentation.
A.PROPER_USER	Mobile device users are not willfully negligent or hostile, and use the device within compliance of a reasonable Enterprise security policy.

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 [MDMPP] and [AGENTMOD].

Table 10: TOE Objectives

Objective	Objective Definition
O.ACCOUNTABILITY	The TOE must provide logging facilities which record management actions undertaken by its administrators.
O.APPLY_POLICY	The TOE must facilitate configuration and enforcement of enterprise security policies on mobile devices via interaction with the mobile OS and the MDM Server. This will include the initial enrollment of the device into management through its entire lifecycle, including policy updates and its possible unenrollment from management services.
O.DATA_PROTECTION_TRANSIT	Data exchanged between the MDM Server and the MDM Agent must be protected from being monitored, accessed, or altered.

O.INTEGRITY	[MDMPP] The TOE will provide the capability to perform self tests to ensure the integrity of critical functionality, software, firmware, and data has been maintained. The TOE will also provide a means to verify the integrity of downloaded updates.
O.MANAGEMENT	[MDMPP] The TOE provides access controls around its management functionality.
O.QUALITY	[MDMPP] To ensure quality of implementation, conformant TOEs leverage services and APIs provided by the runtime environment rather than implementing their own versions of these services and APIs. This is especially important for cryptographic services and other complex operations such as file and media parsing. Leveraging this platform behavior relies upon using only documented and supported APIs.
O.STORAGE	[AGENTMOD] To address the issue of loss of confidentiality of user data in the event of loss of a Mobile Device (T.PHYSICAL), conformant TOEs will use platform provide key storage. The TOE is expected to protect its persistent secrets and private keys.

4.4.2 Security Objectives for the Operational Environment

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

Table 11: Operational Environment Objectives

Objective	Objective Definition
OE.COMPONENTS_RUNNING	[MDMPP] For distributed TOEs the Administrator ensures that the availability of every TOE component is checked as appropriate to reduce the risk of an undetected attack on (or failure of) one or more TOE components. The Administrator also ensures that it is checked as appropriate for every TOE component that the audit functionality is running properly.
OE. PROPER_ADMIN	TOE Administrators are trusted to follow and apply all administrator guidance in a trusted manner.
OE. PROPER_USER	Users of the mobile device are trained to securely use the mobile device and apply all guidance in a trusted manner.
OE.IT_ENTERPRISE	The Enterprise IT infrastructure provides security for a network that is available to the TOE and mobile devices that prevents unauthorized access.
OE.MOBILE_DEVICE_PLATFORM	[AGENTMOD] The MDM Agent relies upon the trustworthy mobile platform and hardware to provide policy enforcement as well as cryptographic services and data protection. The mobile platform provides trusted updates and software integrity verification of the MDM Agent.
OE.TIMESTAMP	[MDMPP] Reliable timestamp is provided by the operational environment for the TOE.
OE.WIRELESS_NETWORK	A wireless network will be available to the mobile devices.

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.

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 underlined text.
- **Iteration operation:** are identified with a number inside parentheses (e.g., "(1)") or a forward-slash with the device type (e.g., "/ANDROID") when functionality differs between device type.

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 Mobile Device Management Protection Profile are denoted by a [MDMPP]; SFRs that originated from the Mobile Device Management Agents PP-Module are denoted by [AGENTMOD].

Table 12: Security Functional Requirements for the TOE

Class Name	Component Identification	Component Name
Security Audit	[MDMPP] FAU_ALT_EXT.1	Server Alerts
	[AGENTMOD] FAU_ALT_EXT.2/ANDROID	Agent Alerts
	[AGENTMOD] FAU_ALT_EXT.2/IOS	Agent Alerts
	[MDMPP] FAU_GEN.1(1)	Audit Data Generation
	[MDMPP] FAU_GEN.1(2)	Audit Generation (MAS Server)
	[AGENTMOD] FAU_GEN.1(2)	Audit Data Generation
	[MDMPP] FAU_NET_EXT.1	Network Reachability Review
	[MDMPP] FAU_SAR.1	Audit Review
	[AGENTMOD] FAU_SEL.1(2)	Security Audit Event Selection
	[MDMPP] FAU_STG_EXT.1	External Trail Storage
Communication	[MDMPP] FCO_CPC_EXT.1	Component Registration Channel Definition
Cryptographic Support	[MDMPP] FCS_CKM.1	Cryptographic Key Generation
	[MDMPP] FCS_CKM.2	Cryptographic Key Establishment
	[MDMPP] FCS_CKM_EXT.4	Cryptographic Key Destruction
	[MDMPP] FCS_COP.1(1)	Cryptographic Operation (Confidentiality Algorithms)

Class Name	Component Identification	Component Name
	[MDMPP] FCS_COP.1(2)	Cryptographic Operation (Hashing Algorithms)
	[MDMPP] FCS_COP.1(3)	Cryptographic Operation (Signature Algorithms)
	[MDMPP] FCS_COP.1(4)	Cryptographic Operation (Keyed-Hash Message Authentication)
	[MDMPP] FCS_RBG_EXT.1	Extended: Random Bit Generation
	[MDMPP] FCS_STG_EXT.1	Cryptographic Key Storage
	[AGENTMOD] FCS_STG_EXT.1(2)	Cryptographic Key Storage
Identification and Authentication	[MDMPP] FIA_ENR_EXT.1/ANDROID	Enrollment of Mobile Device into Management (Android)
	[MDMPP] FIA_ENR_EXT.1/IOS	Enrollment of Mobile Device into Management (iOS)
	[AGENTMOD] FIA_ENR_EXT.2	Agent Enrollment of Mobile Device into Management
	[MDMPP] FIA_UAU.1	Timing of Authentication
	[MDMPP] FIA_X509_EXT.1(1)	Validation of Certificates
	[MDMPP] FIA_X509_EXT.2	X.509 Certification Authentication
	[MDMPP] FIA_X509_EXT.5	X.509 Unique Certificate
Security Management	[MDMPP] FMT_MOF.1(1)	Management of Functions Behavior
	[MDMPP] FMT_MOF.1(2)	Management of Functions Behavior (Enrollment)
	[MDMPP] FMT_MOF.1(3)	Management of Functions in (MAS Server Downloads)
	[MDMPP] FMT_POL_EXT.1	Trusted Policy Update
	[AGENTMOD] FMT_POL_EXT.2	Agent Trusted Policy Update
	[MDMPP] FMT_SMF.1(1)/ANDROID	Specification of Management Functions (Server Configuration of Agent) (Android)
	[MDMPP] FMT_SMF.1(1)/IOS	Specification of Management Functions (Server Configuration of Agent) (iOS)
	[MDMPP] FMT_SMF.1(2)/ANDROID	Specification of Management Functions (Server Configuration of Server) (Android)
	[MDMPP] FMT_SMF.1(2)/IOS	Specification of Management Functions (Server Configuration of Server) (iOS)
	[MDMPP] FMT_SMF.1(3)	Specification of Management Functions (MAS Server)
	[AGENTMOD] FMT_SMF_EXT.4	Specification of Management Functions
	[MDMPP] FMT_SMR.1(1)	Security Management Roles
	[MDMPP] FMT_SMR.1(2)	Security Management Roles (MAS Server)
	[AGENTMOD] FMT_UNR_EXT.1	User Unenrollment Prevention
Protection of the TSF	[MDMPP] FPT_API_EXT.1	Use of Supported Services and API's
	[MDMPP] FPT_ITT.1(2)	Internal TOE TSF Data Transfer (To MDM Agent)

Class Name	Component Identification	Component Name
	[MDMPP] FPT_LIB_EXT.1	Use of Third Party Libraries
	[MDMPP] FPT_TST_EXT.1	Functionality Testing
	[MDMPP] FPT_TUD_EXT.1	Trusted Update
TOE Access	[MDMPP] FTA_TAB.1	Default TOE Access Banners
Trusted Path/Channels	[MDMPP] FTP_ITC_EXT.1	Trusted Channel
	[MDMPP] FTP_ITC.1(1)	Inter-TSF Trusted Channel (Authorized IT Entities)
	[MDMPP] FTP_TRP.1(1)	Trusted Path (for Remote Administration)
	[MDMPP] FTP_TRP.1(2)	Trusted Path (for Enrollment)

6.3 Security Functional Requirements

6.3.1 Class FAU: Security Audit

6.3.1.1 *[MDMPP] FAU_ALT_EXT.1 Server Alerts*

FAU_ALT_EXT.1.1

The TSF shall alert the administrators in the event of any of the following:

a. Change in enrollment status
b. Failure to apply policies to a mobile device
c. [[*presence of blacklisted apps, presence of non-whitelisted apps, absence of required apps, compromised (jailbroken or rooted) device, last time a device communicated with the MDM Server, unapproved model (iOS only), unapproved device manufacturer (Android only), unapproved operating system version*]]

6.3.1.2 *[AGENTMOD] FAU_ALT_EXT.2/ANDROID Agent Alerts*

FAU_ALT_EXT.2.1/ANDROID

The MDM Agent shall provide an alert via the trusted channel to the MDM Server in the event of any of the following audit events:

- successful application of policies to a mobile device,
- [generating] periodic reachability events,
- [
  - change in enrollment state,
  - failure to install an application from the MAS Server,
  - failure to update an application from the MAS Server,
  - [*detection of blacklisted apps, detection of non-whitelisted apps, required apps missing, jailbroken or rooted device, unapproved device manufacturer, unapproved operating system version*]].

FAU_ALT_EXT.2.2/ANDROID

The MDM Agent shall queue alerts if the trusted channel is not available.

6.3.1.3 *[AGENTMOD] FAU_ALT_EXT.2/IOS Agent Alerts*

FAU_ALT_EXT.2.1/IOS

The MDM Agent shall provide an alert via the trusted channel to the MDM Server in the event of any of the following audit events:

- successful application of policies to a mobile device,
- [receiving] periodic reachability events,
- [
  - change in enrollment state,
  - failure to install an application from the MAS Server,
  - failure to update an application from the MAS Server,
  - [*detection of blacklisted apps, detection of non-whitelisted apps, required apps missing, jailbroken or rooted device, unapproved model, unapproved operating system version*]].

FAU_ALT_EXT.2.2/IOS

The MDM Agent shall queue alerts if the trusted channel is not available.

6.3.1.4 *[MDMPP] FAU_GEN.1(1) Audit Data Generation*

FAU_GEN.1.1(1)

The TSF shall [invoke platform-provided functionality, implement functionality] to generate an audit record of the following auditable events:

a. Start-up and shutdown of the MDM System
b. All administrative actions
c. [Commands issued to the MDM Agent]
d. Specifically defined auditable events listed in **Table 13**
e. [[*MDM Agent alerts (generated by FAU_ALT_EXT.2.1 in the MDM Agent PP-Module), MDM Agent audit records (generated by FAU_GEN.1.1(2) in the MDM Agent PP-Module)*]].

FAU_GEN.1.2(1)

The TSF shall record within each TSF audit record at least the following information:

- date and time of the event
- type of event
- subject identity
- (if relevant) the outcome (success or failure) of the event
- additional information in **Table 13**
- [*no other information*].

Table 13: Server Auditable Events

Requirement	Auditable Events	Additional Audit Record Contents
FAU_ALT_EXT.1 (man)	Type of alert.	Identity of Mobile Device that sent alert.
FAU_GEN.1(1) (man)	None.	N/A
FAU_GEN.1(2) (sel)	None.	N/A
FAU_NET_EXT.1 (man)	None.	N/A
FAU_SAR.1 (opt)	None.	N/A
FAU_STG_EXT.1 (man)	None.	N/A
FCO_CPC_EXT.1 (obj)	Enabling/Disabling communications between a pair of components.	Identities of the endpoints pairs enabled or disabled.
FCS_CKM_EXT.4 (man)	None.	N/A
FCS_CKM.1 (man)	[Failure of key generation activity for authentication keys]	No additional information.
FCS_CKM.2 (man)	None.	N/A
FCS_COP.1(1) (man)	None.	N/A
FCS_COP.1(2) (man)	None.	N/A
FCS_COP.1(3) (man)	None.	N/A
FCS_COP.1(4) (man)	None.	N/A
FCS_RBG_EXT.1 (man)	Failure of the randomization process.	No additional information.
FCS_STG_EXT.1 (man)	None.	N/A
FIA_ENR_EXT.1/ANDROID (man)	Failure of MD user authentication.	Presented username.
FIA_ENR_EXT.1/IOS (man)	Failure of MD user authentication.	Presented username.
FIA_UAU.1 (man)	None.	N/A
FIA_X509_EXT.1(1) (man)	Failure to validate X.509 certificate.	Reason for failure.
FIA_X509_EXT.2 (man)	Failure to establish connection to determine revocation status.	No additional information.

FIA_X509_EXT.5 (man)	None.	N/A
FMT_MOF.1(1) (man)	Issuance of command to perform function. Change of policy settings.	Command sent and identity of MDM Agent recipient(s). Policy changed and value or full policy.
FMT_MOF.1(2) (man)	Enrollment by a user.	Identity of user.
FMT_MOF.1(3) (sel)	None.	N/A
FMT_POL_EXT.1 (man)	None.	N/A
FMT_SMF.1(1)/ANDROID (man)	None.	N/A
FMT_SMF.1(1)/IOS (man)	None.	N/A
FMT_SMF.1(2)/ANDROID (man)	Success or failure of function.	No additional information.
FMT_SMF.1(2)/IOS (man)	Success or failure of function.	No additional information.
FMT_SMF.1(3) (sel)	None.	N/A
FMT_SMR.1(1) (man)	None.	N/A
FMT_SMR.1(2) (sel)	None.	N/A
FPT_API_EXT.1 (man)	None.	N/A
FPT_ITT.1(2) (sel)	Initiation and termination of the trusted channel.	Trusted channel protocol. Identity of initiator and recipient.
FPT_LIB_EXT.1 (man)	None.	N/A
FPT_TST_EXT.1 (man)	Initiation of self-test. Failure of self-test. Detected integrity violation.	Algorithm that caused failure. The TSF code file that caused the integrity violation.
FPT_TUD_EXT.1 (man)	Success or failure of signature verification.	No additional information.
FTA_TAB.1 (opt)	Change in banner setting.	No additional information.
FTP_ITC.1(1) (man)	Initiation and termination of the trusted channel.	Trusted channel protocol. Non-TOE endpoint of connection.
FTP_ITC_EXT.1 (man)	None.	N/A
FTP_TRP.1(1) (man)	Initiation and termination of the trusted channel.	Trusted channel protocol. Identity of administrator.
FTP_TRP.1(2) (man)	Initiation and termination of the trusted channel.	Trusted channel protocol.

6.3.1.5 *[MDMPP] FAU_GEN.1(2) Audit Generation (MAS Server)*

FAU_GEN.1.1(2)

The MAS Server shall be able to generate an audit record of the following auditable events:

a. Failure to push a new application on a managed mobile device
b. Failure to update an existing application on a managed mobile device.

FAU_GEN.1.2(2)

The [MAS Server] shall record within each TSF audit record at least the following information:

- date and time of the event
- type of event
- mobile device identity
- [*no other information*].

6.3.1.6 *[AGENTMOD] FAU_GEN.1(2) Audit Data Generation*

FAU_GEN.1.1(2)

The MDM Agent shall [invoke platform-provided functionality, implement functionality] to generate an MDM Agent audit record of the following auditable events:

a. Start-up and shutdown of the MDM Agent;
b. All auditable events for not specified level of audit; and
c. MDM policy updated, any modification commanded by the MDM Server, specifically defined auditable events listed in **Table 14**, and [no other events].

FAU_GEN.1.2(2)

The [TSF] shall record within each MDM Agent audit record at least the following information:

a. Date and time of the event, type of event, subject identity, (if relevant) the outcome (success or failure) of the event, additional information in **Table 14**; and
b. For each audit event type, based on the auditable event definitions of the functional components included in the PP-Module/ST, [*no other information*].

Table 14: Agent Auditable Events

Requirement	Auditable Events	Additional Audit Record Contents
FAU_ALT_EXT.2/ANDROID	Success/failure of sending alert.	No additional information.
FAU_ALT_EXT.2/IOS	Success/failure of sending alert.	No additional information.
FAU_GEN.1	None.	N/A
FAU_SEL.1	All modifications to the audit configuration that occur while	No additional information.

	the audit collection functions are operating.	
FCS_STG_EXT.1(2)	None.	N/A
FIA_ENR_EXT.2	Enrollment in management.	Reference identifier of MDM Server.
FMT_POL_EXT.2	Failure of policy validation.	Reason for failure of validation.
FMT_SMF_EXT.4	Outcome (Success/failure) of function.	No additional information.
FMT_UNR_EXT.1.1	[None]	No additional information.

6.3.1.7 *[MDMPP] FAU_NET_EXT.1 Network Reachability Review*

FAU_NET_EXT.1.1

The TSF shall provide authorized administrators with the capability to read the network connectivity status of an enrolled agent.

6.3.1.8 *[MDMPP] FAU_SAR.1 Audit Review*

FAU_SAR.1.1

The TSF shall [invoke platform-provided functionality, implement functionality] to provide Authorized Administrators with the capability to read all audit data from the audit records.

FAU_SAR.1.2

The TSF shall [invoke platform-provided functionality, implement functionality] to provide the audit records in a manner suitable for the Authorized Administrators to interpret the information.

6.3.1.9 *[AGENTMOD] FAU_SEL.1(2) Security Audit Event Selection*

FAU_SEL.1.1(2)

The TSF shall [invoke platform-provided functionality, implement functionality] to select the set of events to be audited from the set of all auditable events based on the following attributes:

a. event type
b. success of auditable security events, failure of auditable security events, [*no other attributes*].

6.3.1.10 *[MDMPP] FAU_STG_EXT.1 External Trail Storage*

FAU_STG_EXT.1.1

The TSF shall be able to use a trusted channel per FTP_ITC.1(1) to transmit audit data to an external IT entity and [no other method].

6.3.2 Class FCO: Communication

6.3.2.1 *[MDMPP] FCO_CPC_EXT.1 Component Registration Channel Definition*

FCO_CPC_EXT.1.1

The TSF shall [implement functionality] to require an Administrator to enable communications between any pair of TOE components before such communication can take place.

FCO_CPC_EXT.1.2[1]

The TSF shall [invoke platform-provided functionality] to implement a registration process in which components establish and use a communications channel that uses [

- A channel that meets the secure channel requirements in [FTP_TRP.1(2)]

] for at least TSF data.

FCO_CPC_EXT.1.3

The TSF shall [implement functionality] to enable an Administrator to disable communications between any pair of TOE components.

6.3.3 Class FCS: Cryptographic Support

6.3.3.1 *[MDMPP] FCS_CKM.1 Cryptographic Key Generation*

FCS_CKM.1.1

The TSF 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 meets FIPS PUB 186-4, “Digital Signature Standard (DSS)”, Appendix B.3;
- ECC schemes using “NIST curves” P-384 and [P-256] that meet the following: FIPS PUB 186-4, “Digital Signature Standard (DSS)”, Appendix B.4].

6.3.3.2 *[MDMPP] FCS_CKM.2 Cryptographic Key Establishment*

FCS_CKM.2.1

The TSF 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: RSAES-PKCS1-v1_5 as specified in Section 7.2 of RFC 8017, "Public-Key Cryptography Standards (PKCS) #1: RSA Cryptography Specifications Version 2.1",

[1] FTP_TRP.1(2) was added as a selection in FCO_CPC_EXT.1.2 through Technical Rapid Response Team (TRRT) response to Technical Query 869

- 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"].

6.3.3.3 *[MDMPP] FCS_CKM_EXT.4 Cryptographic Key Destruction*

FCS_CKM_EXT.4.1

The TSF shall destroy plaintext keying material and critical security parameters by: [

- invoking platform-provided functionality with the following rules:
  - For volatile memory, the destruction shall be executed by [
    - a single direct overwrite consisting of [zeroes]]
  - For non-volatile memory that consists of the invocation of an interface provided by the underlying platform that [
    - logically addresses the storage location of the key and performs a [single] direct overwrite consisting of [ones]].

FCS_CKM_EXT.4.2

The TSF shall destroy all plaintext keying material and critical security parameters (CSPs) when no longer needed.

6.3.3.4 *[MDMPP] FCS_COP.1(1) Cryptographic Operation (Confidentiality Algorithms)*

FCS_COP.1.1(1)

The TSF shall [invoke platform-provided functionality] to perform encryption/decryption in accordance with a specified cryptographic algorithm [

- AES-CBC (as defined in FIPS PUB 197, and NIST SP 800-38A) mode,
- AES-GCM (as defined in NIST SP 800-38D)]

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

6.3.3.5 *[MDMPP] FCS_COP.1(2) Cryptographic Operation (Hashing Algorithms)*

FCS_COP.1.1(2)

The TSF shall [invoke platform-provided functionality, implements functionality] to perform cryptographic hashing 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.

6.3.3.6 *[MDMPP] FCS_COP.1(3) Cryptographic Operation (Signature Algorithms)*

FCS_COP.1.1(3)

The TSF shall [invoke platform-provided functionality, implements functionality] to perform cryptographic signature services (generation and verification) in accordance with a specified cryptographic algorithm [

- RSA schemes using cryptographic key sizes of 2048-bit or greater that meet the following: FIPS PUB 186-4, "Digital Signature Standard (DSS)", Section 4,
- ECDSA schemes using "NIST curves" P-384 and [P-256, P-521] that meet the following: FIPS PUB 186-4, "Digital Signature Standard (DSS)", Section 5].

6.3.3.7 *[MDMPP] FCS_COP.1(4)* *Cryptographic Operation (Keyed-Hash Message Authentication)*

FCS_COP.1.1(4)

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

6.3.3.8 *[MDMPP] FCS_RBG_EXT.1* *Extended: Random Bit Generation*

FCS_RBG_EXT.1.1

The TSF shall [invoke platform-provided functionality] to perform all deterministic random bit generation services in accordance with NIST Special Publication 800-90A using [CTR_DRBG (AES)].

FCS_RBG_EXT.1.2

The deterministic RBG shall be seeded by an entropy source that accumulates entropy from [a platform-based RBG] 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.

6.3.3.9 *[MDMPP] FCS_STG_EXT.1* *Cryptographic Key Storage*

FCS_STG_EXT.1.1

The TSF shall utilize [platform-provided key storage] for all persistent secrets and private keys.

6.3.3.10 *[AGENTMOD] FCS_STG_EXT.1(2) Cryptographic Key Storage*

FCS_STG_EXT.1.1(2)

The MDM Agent shall use the platform-provided key storage for all persistent secret and private keys.

6.3.4 Class FIA: Identification and Authentication

6.3.4.1 *[MDMPP] FIA_ENR_EXT.1/ANDROID* *Enrollment of Mobile Device into Management (Android)*

FIA_ENR_EXT.1.1/ANDROID

The TSF shall authenticate the remote users over a trusted channel during the enrollment of a mobile device.

FIA_ENR_EXT.1.2/ANDROID

The TSF shall limit the user's enrollment of devices to devices specified by [IMEI] and [specific device models, a number of devices, [*serial number, manufacturer, operating system*]].

6.3.4.2 *[MDMPP] FIA_ENR_EXT.1/IOS Enrollment of Mobile Device into Management (iOS)*

FIA_ENR_EXT.1.1/IOS

The TSF shall authenticate the remote users over a trusted channel during the enrollment of a mobile device.

FIA_ENR_EXT.1.2/IOS

The TSF shall limit the user's enrollment of devices to devices specified by [[*DEP identifier*]] and [no other features].

6.3.4.3 *[AGENTMOD] FIA_ENR_EXT.2 Agent Enrollment of Mobile Device into Management*

FIA_ENR_EXT.2.1

The MDM Agent shall record the reference identifier of the MDM Server during the enrollment process.

6.3.4.4 *[MDMPP] FIA_UAU.1 Timing of Authentication*

FIA_UAU.1.1

The TSF shall [implement functionality] to allow [*view login banner, view MDM software version, request password reset*] on behalf of the user to be performed before the user is authenticated with the Server.

FIA_UAU.1.2

The TSF shall [implement functionality] that requires each user to be successfully authenticated with the Server before allowing any other TSF-mediated actions on behalf of that user.

6.3.4.5 *[MDMPP] FIA_X509_EXT.1(1) Validation of Certificates*

FIA_X509_EXT.1.1(1)[2]

The TSF shall [invoke platform-provided functionality, 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 TSF 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 TSF shall validate the revocation status of the certificate using [the Online Certificate Status Protocol (OCSP) as specified in RFC 2560].

[2] TD0467

- The TSF 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.
  - CSP 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(1)

The TSF shall [invoke platform-provided functionality, implement functionality] to treat a certificate as a CA certificate only if the basicConstraints extension is present and the CA flag is set to TRUE.

6.3.4.6 *[MDMPP] FIA_X509_EXT.2 X.509 Certificate Authentication*

FIA_X509_EXT.2.1

The TSF shall [

- invoke platform-provided functionality to use X.509v3 certificates as defined by RFC 5280 to support authentication for [HTTPS, TLS], and [
  - code signing for system software updates,
  - code signing for integrity verification,
  - policy signing],
- implement functionality to use X.509v3 certificates as defined by RFC 5280 to support authentication for [
  - no protocols]

  and [
  - policy signing].]

FIA_X509_EXT.2.2

When the [TSF, TOE platform] cannot establish a connection to determine the validity of a certificate, the TSF shall [invoke platform-provided functionality, implement functionality] to [not accept the certificate].

6.3.4.7 *[MDMPP] FIA_X509_EXT.5 X.509 Unique Certificate*

FIA_X509_EXT.5.1

The TSF shall [invoke platform-provided functionality] to require a unique certificate for each client device.

6.3.5 Class FMT: Security Management

6.3.5.1 *[MDMPP] FMT_MOF.1(1) Management of Functions Behavior*

FMT_MOF.1.1(1)

The TSF shall restrict the ability to perform the functions

- listed in FMT_SMF.1(1)
- enable, disable, and modify policies listed in FMT_SMF.1(1)
- listed in FMT_SMF.1(2)
- [enable, disable and modify policies listed in FMT_SMF.1(3)]

to authorized administrators.

6.3.5.2 *[MDMPP] FMT_MOF.1(2) Management of Functions Behavior (Enrollment)*

FMT_MOF.1.1(2)

The MDM Server shall restrict the ability to initiate the enrollment process to authorized administrators and MD users.

6.3.5.3 *[MDMPP] FMT_MOF.1(3) Management of Functions in (MAS Server Downloads)*

FMT_MOF.1.1(3)

The MAS Server shall restrict the ability to download applications, allowing only enrolled mobile devices that are compliant with MDM policies and assigned to a user in the application access group to perform this function.

6.3.5.4 *[MDMPP] FMT_POL_EXT.1 Trusted Policy Update*

FMT_POL_EXT.1.1

The TSF shall provide digitally signed policies and policy updates to the MDM Agent.

6.3.5.5 *[AGENTMOD] FMT_POL_EXT.2 Agent Trusted Policy Update*

FMT_POL_EXT.2.1

The MDM Agent shall only accept policies and policy updates that are digitally signed by a certificate that has been authorized for policy updates by the MDM Server.

FMT_POL_EXT.2.2

The MDM Agent shall not install policies if the policy-signing certificate is deemed invalid.

6.3.5.6 *[MDMPP] FMT_SMF.1(1)/ANDROID Specification of Management Functions (Server configuration of Agent) (Android)*

FMT_SMF.1.1(1)/ANDROID

The MDM Server shall be capable of communicating the following commands to the MDM Agent:

1. transition to the locked state (MDF Function 6)
2. full wipe of protected data (MDF Function 7)
3. unenroll from management
4. install policies
5. query connectivity status
6. query the current version of the MD firmware/software
7. query the current version of the hardware model of the device
8. query the current version of installed mobile applications
9. import X.509v3 certificates into the Trust Anchor Database (MDF Function 11)
10. install applications (MDF Function 16)
11. update system software (MDF Function 15)
12. remove applications (MDF Function 14)

and the following commands to the MDM Agent: [

- 13. remove Enterprise applications (MDF Function 17),
- 14. wipe Enterprise data (MDF Function 28),
- 15. remove imported X.509v3 certificates and [
    - no other X.509v3 certificates

  ] in the Trust Anchor Database (MDF Function 12),
- 16. alert the user,
- 22. place applications into application process groups based on [*enterprise ownership*] (MDF Function 43)]

and the following MD configuration policies:

25. password policy:
    a. minimum password length
    b. minimum password complexity
    c. maximum password lifetime (MDF Function 1)
26. session locking policy:
    a. screen-lock enabled/disabled
    b. screen lock timeout
    c. number of authentication failures (MDF Function 2)
27. wireless networks (SSIDs) to which the MD may connect (MDF Function 2)
28. security policy for each wireless network:
    a. [
        - specify the CA(s) from which the MD will accept WLAN authentication server certificate(s)]
    b. ability to specify security type
    c. ability to specify authentication protocol
    d. specify the client credentials to be used for authentication
    e. [*no other WLAN management functions*] (WLAN Client Function 1)
29. application installation policy by [
    - specifying authorized application repository(s),

- denying application installation], (MDF Function 8)

30. enable/disable policy for [*camera, microphone, screen capture*] across device, [
    - and no other method], (MDF Function 5)

and the following MD configuration policies: [

- 31. enable/disable policy for the VPN protection across MD, [
    - no other method] (MDF Function 3),
- 32. enable/disable policy for [*Wi-Fi, cellular, Bluetooth, NFC*], (MDF Function 4),
- 33. enable/disable policy for data signaling over [*USB, headphone jack, removable storage card (SD card)*], (MDF Function 24),
- 34. enable/disable policy for [*Wi-Fi tethering, USB tethering, and Bluetooth tethering*], (MDF Function 25),
- 35. enable/disable policy for developer modes, (MDF Function 26),
- 36. enable policy for data-at rest protection, (MDF Function 20),
- 37. enable policy for removable media's data-at-rest protection, (MDF Function 21),
- 40. enable/disable policy for display notification in the locked state of [
    - email notifications,
    - calendar appointments,
    - contact associated with phone call notification,
    - text message notification,
    - other application-based notifications

  ] (MDF Function 19),
- 47. the unlock banner policy, (MDF Function 36),
- 49. enable/disable [
    - USB mass storage mode] (MDF Function 39),
- 50. enable/disable backup of [
    - all applications,
    - selected applications,
    - selected groups of applications,
    - configuration data

  ] to [locally connected system, remote system] (MDF Function 40),
- 52. enable/disable location services: [
    - across device,
    - no other method] (MDF Function 22),
- 53. enable/disable policy for user unenrollment,
- 54. enable/disable policy for the Always-On VPN protection across device (MDF Function 45),
- 55. enable/disable policy for use of Biometric Authentication Factor (MDF Function 23),
- 58. enable/disable automatic updates of system software,
- 59. enable/disable removable media
- 60. [*application installation policy by [*
    - *specifying a set of allowed applications (an application whitelist)]*]]

6.3.5.7 *[MDMPP] FMT_SMF.1(1)/IOS* *Specification of Management Functions (Server configuration of Agent) (iOS)*

FMT_SMF.1.1(1)/IOS

The MDM Server shall be capable of communicating the following commands to the MDM Agent:

1. transition to the locked state (MDF Function 6)
2. full wipe of protected data (MDF Function 7)
3. unenroll from management
4. install policies
5. query connectivity status
6. query the current version of the MD firmware/software
7. query the current version of the hardware model of the device
8. query the current version of installed mobile applications
9. import X.509v3 certificates into the Trust Anchor Database (MDF Function 11)
10. install applications (MDF Function 16)
11. update system software (MDF Function 15)
12. remove applications (MDF Function 14)

and the following commands to the MDM Agent: [

- 13. remove Enterprise applications (MDF Function 17),
- 14. wipe Enterprise data (MDF Function 28),
- 15. remove imported X.509v3 certificates and [
  - no other X.509v3 certificates

  ] in the Trust Anchor Database (MDF Function 12),
- 16. alert the user,
- 22. place applications into application process groups based on [*enterprise ownership*] (MDF Function 43),
- 23. revoke Biometric template]

and the following MD configuration policies:

25. password policy:
    a. minimum password length
    b. minimum password complexity
    c. maximum password lifetime (MDF Function 1)
26. session locking policy:
    a. screen-lock enabled/disabled
    b. screen lock timeout
    c. number of authentication failures (MDF Function 2)
27. wireless networks (SSIDs) to which the MD may connect (MDF Function 2)
28. security policy for each wireless network:
    a. [
       - specify the CA(s) from which the MD will accept WLAN authentication server certificate(s),

- specify the FQDN(s) of acceptable WLAN authentication server certificate(s)]

b. ability to specify security type
c. ability to specify authentication protocol
d. specify the client credentials to be used for authentication
e. [*no other WLAN management functions*] (WLAN Client Function 1)

29. application installation policy by [
    - specifying authorized application repository(s)], (MDF Function 8)
30. enable/disable policy for [*camera, screen capture*] across device, [
    - and no other method], (MDF Function 5)

and the following MD configuration policies: [

- 31. enable/disable policy for the VPN protection across MD
  - [on a per-app basis,
  - no other method] (MDF Function 3),
- 36. enable policy for data-at rest protection, (MDF Function 20),
- 40. enable/disable policy for display notification in the locked state of [
  - email notifications,
  - calendar appointments,
  - contact associated with phone call notification,
  - text message notification,
  - other application-based notifications

  ] (MDF Function 19),
- 47. the unlock banner policy, (MDF Function 36),
- 50. enable/disable backup of [
  - all applications,
  - configuration data

  ] to [remote system] (MDF Function 40),
- 55. enable/disable policy for use of Biometric Authentication Factor (MDF Function 23),
- 60. [*application installation policy by [*
  - *specifying a set of allowed applications (an application whitelist)]*]
- **61.** [*iOS Hub Agent passcode authentication policy:*
  - *Minimum Passcode Length*]]

6.3.5.8 *[MDMPP] FMT_SMF.1(2)/ANDROID* *Specification of Management Functions (Server Configuration of Server) (Android)*

FMT_SMF.1.1(2)/ANDROID

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

a. choose X.509v3 certificates for MDM Server use
b. configure the [
  - devices specified by [IMEI, [*serial number*]],
  - specific device models,

- a number of devices

] and [[*manufacturer, operating system*]] allowed for enrollment,

c. [

2. configure the TOE unlock banner,
3. configure periodicity of the following commands to the agent: [*query connectivity status, query the current version of the MD firmware/software, query the current version of the hardware model of the device, query the current version of installed mobile applications*],
4. configure the privacy-sensitive information that will and will not be collected from particular mobile devices]
6. configure the interaction between TOE components
8. [*configure server administrator login session timeout, configure Enterprise certificate to be used for signing policies, configure MDM Agent/platform to perform a network reachability test, configure transfer of MDM server logs to another server for storage, analysis, and reporting*]].

6.3.5.9 *[MDMPP] FMT_SMF.1(2)/IOS Specification of Management Functions (Server Configuration of Server) (iOS)*

FMT_SMF.1.1(2)/IOS

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

a. choose X.509v3 certificates for MDM Server use
b. configure the [

- devices specified by [[*DEP identifier*]],

] and [no other features] allowed for enrollment

c. [

2. configure the TOE unlock banner,
3. configure periodicity of the following commands to the agent: [*query connectivity status, query the current version of the MD firmware/software, query the current version of the hardware model of the device, query the current version of installed mobile applications*],
4. configure the privacy-sensitive information that will and will not be collected from particular mobile devices
6. configure the interaction between TOE components
8. [*configure server administrator login session timeout, configure Enterprise certificate to be used for signing policies, configure MDM Agent/platform to perform a network reachability test, configure transfer of MDM server logs to another server for storage, analysis, and reporting*]].

6.3.5.10 *[MDMPP] FMT_SMF.1(3) Specification of Management Functions (MAS Server)*

FMT_SMF.1.1(3)

The MAS Server shall be capable of performing the following management functions:

a. Configure application access groups
b. Download applications
c. [no other functions]

6.3.5.11 *[AGENTMOD] FMT_SMF_EXT.4 Specification of Management Functions*

FMT_SMF_EXT.4.1

The MDM Agent shall be capable of interacting with the platform to perform the following functions:

- Import the certificates to be used for authentication of MDM Agent communications,
- [administrator-provided device management functions in MDM PP]
- [no additional functions].

FMT_SMF_EXT.4.2

The MDM Agent shall be capable of performing the following functions:

- Enroll in management
- Configure whether users can unenroll from management
- [configure periodicity of reachability events].

6.3.5.12 *[MDMPP] FMT_SMR.1(1) Security Management Roles*

FMT_SMR.1.1(1)

The TSF shall maintain the roles administrator, MD user, and [[*server primary administrator*, *security configuration administrator*, *device user group administrator*, *auditor*]].

FMT_SMR.1.2(1)

The TSF shall be able to associate users with roles.

6.3.5.13 *[MDMPP] FMT_SMR.1(2) Security Management Roles (MAS Server)*

FMT_SMR.1.1(2)

The TSF shall additionally maintain the roles enrolled mobile devices, application access groups, and [*server primary administrator, security configuration administrator, device user group administrator, auditor*].

FMT_SMR.1.2(2)

The MAS Server shall be able to associate users with roles.

6.3.5.14 *[AGENTMOD] FMT_UNR_EXT.1 User Unenrollment Prevention*

FMT_UNR_EXT.1.1

The MDM Agent shall provide a mechanism to enforce the following behavior upon an attempt to unenroll the mobile device from management: [prevent the unenrollment from occurring].

6.3.6 Class FPT: Protection of the TSF

6.3.6.1 *[MDMPP] FPT_API_EXT.1 Use of Supported Services and APIs*

FPT_API_EXT.1.1

The TSF shall use only documented platform API's.

6.3.6.2 *[MDMPP] FPT_ITT.1(2) Internal TOE TSF Data Transfer (To MDM Agent)*

FPT_ITT.1.1(2)

The TSF shall [

- invoke platform-provided functionality to use [
  - mutually authenticated TLS,
  - HTTPS]

] to protect all data from disclosure and modification when it is transferred between TSF and MDM Agent.

6.3.6.3 *[MDMPP] FPT_LIB_EXT.1 Use of Third Party Libraries*

FPT_LIB_EXT.1.1

The MDM software shall be packaged with only [*a list of third-party libraries in Appendix A*].

6.3.6.4 *[MDMPP] FPT_TST_EXT.1 Functionality Testing*

FPT_TST_EXT.1.1

The TSF shall run a suite of self tests during initial start-up (power on) to demonstrate correct operation of the TSF.

FPT_TST_EXT.1.2

The TSF shall [invoke platform-provided functionality] to provide the capability to verify the integrity of stored TSF executable code when it is loaded for execution through the use of the [TOE platform]-provided cryptographic services.

6.3.6.5 *[MDMPP] FPT_TUD_EXT.1 Trusted Update*

FPT_TUD_EXT.1.1[3]

The TSF shall provide Authorized Administrators the ability to query the current version of the software.

[3] TD0438

FPT_TUD_EXT.1.2

The TSF shall [invoke platform-provided functionality] to provide Authorized Administrators the ability to initiate updates to TSF software.

FPT_TUD_EXT.1.3

The TSF shall [invoke platform-provided functionality] to provide a means to verify software updates to the TSF using a digital signature mechanism prior to installing those updates.

6.3.7 Class FTA: TOE Access

6.3.7.1 *[MDMPP] FTA_TAB.1* *Default TOE Access Banners*

FTA_TAB.1.1

Before establishing a user session, the TSF shall [implement functionality] to display an Administrator-specified advisory notice and consent warning message regarding use of the TOE.

6.3.8 Class FTP: Trusted Path/Channels

6.3.8.1 *[MDMPP] FTP_ITC_EXT.1* *Trusted Channel*

FTP_ITC_EXT.1.1

The TSF shall provide a communication channel between itself and [

- an MDM Agent that is internal to the TOE

] that is logically distinct from other communication channels, as specified in [FPT_ITT.1(2)].

6.3.8.2 *[MDMPP] FTP_ITC.1(1)* *Inter-TSF Trusted Channel (Authorized IT Entities)*

FTP_ITC.1.1(1)

The TSF shall [

- invoke platform-provided functionality to use [
  - mutually authenticated TLS,
  - HTTPS]

] to provide a trusted communication channel between itself and authorized IT entities supporting the following capabilities: audit server, [authentication server] that is logically distinct from other communication channels and provides assured identification of its end points and protection of the channel data from modification or disclosure.

FTP_ITC.1.2(1)

The TSF shall [invoke platform-provided functionality] to permit the MDM Server or other authorized IT entities to initiate communication via the trusted channel.

FTP_ITC.1.3(1)

The TSF shall [invoke platform-provided functionality] to initiate communication via the trusted channel for [*sending audit data to the Syslog Server, sending authentication requests to LDAP*].

6.3.8.3 *[MDMPP] FTP_TRP.1(1) Trusted Path (for Remote Administration)*

FTP_TRP.1.1(1)

The TSF shall [

- invoke platform-provided functionality to use [
  - TLS,
  - HTTPS]

] to provide a trusted communication path between itself as a [server] and remote administrators that is logically distinct from other communication paths and provides assured identification of its endpoints and protection of the communicated data from modification, disclosure.

FTP_TRP.1.2(1)

The TSF shall [invoke platform-provided functionality] to permit remote administrators to initiate communication via the trusted path.

FTP_TRP.1.3(1)

The TSF shall [invoke platform-provided functionality] to require the use of the trusted path for all remote administration actions.

6.3.8.4 *[MDMPP] FTP_TRP.1(2) Trusted Path (for Enrollment)*

FTP_TRP.1.1(2)

The TSF shall [

- invoke platform-provided functionality to use [
  - TLS,
  - HTTPS]

] to provide a trusted communication path between itself (as a server) and MD users that is logically distinct from other communication paths and provides assured identification of its endpoints and protection of the communicated data from disclosure and detection of modification of the communicated data from modification, disclosure.

FTP_TRP.1.2(2)

The TSF shall [invoke platform-provided functionality] to permit MD users to initiate communication via the trusted path.

FTP_TRP.1.3(2)

The TSF shall [invoke platform-provided functionality] to require the use of the trusted path for all MD user actions.

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 and a subset of the selection-based and 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.

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 [MDMPP] and [AGENTMOD].

7.1 Class ASE: Security Target evaluation

7.1.1 ST introduction (ASE_INT.1)

7.1.1.1 *Developer action elements:*

ASE_INT.1.1D

The developer shall provide an ST introduction.

7.1.1.2 *Content and presentation elements:*

ASE_INT.1.1C

The ST introduction shall contain an ST reference, a TOE reference, a TOE overview and a TOE description.

ASE_INT.1.2C

The ST reference shall uniquely identify the ST.

ASE_INT.1.3C

The TOE reference shall uniquely identify the TOE.

ASE_INT.1.4C

The TOE overview shall summarise the usage and major security features of the TOE.

ASE_INT.1.5C

The TOE overview shall identify the TOE type.

ASE_INT.1.6C

The TOE overview shall identify any non-TOE hardware/software/firmware required by the TOE.

ASE_INT.1.7C

The TOE description shall describe the physical scope of the TOE.

ASE_INT.1.8C

The TOE description shall describe the logical scope of the TOE.

7.1.1.3 *Evaluator action elements:*

ASE_INT.1.1E

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

ASE_INT.1.2E

The evaluator shall confirm that the TOE reference, the TOE overview, and the TOE description are consistent with each other.

7.1.2 Conformance claims (ASE_CCL.1)

7.1.2.1 *Developer action elements:*

ASE_CCL.1.1D

The developer shall provide a conformance claim.

ASE_CCL.1.2D

The developer shall provide a conformance claim rationale

7.1.2.2 *Content and presentation elements:*

ASE_CCL.1.1C

The conformance claim shall contain a CC conformance claim that identifies the version of the CC to which the ST and the TOE claim conformance.

ASE_CCL.1.2C

The CC conformance claim shall describe the conformance of the ST to CC Part 2 as either CC Part 2 conformant or CC Part 2 extended.

ASE_CCL.1.3C

The CC conformance claim shall describe the conformance of the ST to CC Part 3 as either CC Part 3 conformant or CC Part 3 extended.

ASE_CCL.1.4C

The CC conformance claim shall be consistent with the extended components definition.

ASE_CCL.1.5C

The conformance claim shall identify all PPs and security requirement packages to which the ST claims conformance.

ASE_CCL.1.6C

The conformance claim shall describe any conformance of the ST to a package as either package-conformant or package-augmented.

ASE_CCL.1.7C

The conformance claim rationale shall demonstrate that the TOE type is consistent with the TOE type in the PPs for which conformance is being claimed.

ASE_CCL.1.8C

The conformance claim rationale shall demonstrate that the statement of the security problem definition is consistent with the statement of the security problem definition in the PPs for which conformance is being claimed.

ASE_CCL.1.9C

The conformance claim rationale shall demonstrate that the statement of security objectives is consistent with the statement of security objectives in the PPs for which conformance is being claimed.

ASE_CCL.1.10C

The conformance claim rationale shall demonstrate that the statement of security requirements is consistent with the statement of security requirements in the PPs for which conformance is being claimed.

7.1.2.3 *Evaluator action elements:*

ASE_CCL.1.1E

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

7.1.3 Security objectives for the operational environment (ASE_OBJ.1)

7.1.3.1 *Developer action elements:*

ASE_OBJ.1.1D

The developer shall provide a statement of security objectives.

7.1.3.2 *Content and presentation elements:*

ASE_OBJ.1.1C

The statement of security objectives shall describe the security objectives for the operational environment.

7.1.3.3 *Evaluator action elements:*

ASE_OBJ.1.1E

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

7.1.4 Extended components definition (ASE_ECD.1)

7.1.4.1 *Developer action elements:*

ASE_ECD.1.1D

The developer shall provide a statement of security requirements.

ASE_ECD.1.2D

The developer shall provide an extended components definition.

7.1.4.2 *Content and presentation elements:*

ASE_ECD.1.1C

The statement of security requirements shall identify all extended security requirements.

ASE_ECD.1.2C

The extended components definition shall define an extended component for each extended security requirement.

ASE_ECD.1.3C

The extended components definition shall describe how each extended component is related to the existing CC components, families, and classes.

ASE_ECD.1.4C

The extended components definition shall use the existing CC components, families, classes, and methodology as a model for presentation.

ASE_ECD.1.5C

The extended components shall consist of measurable and objective elements such that conformance or nonconformance to these elements can be demonstrated.

7.1.4.3 *Evaluator action elements:*

ASE_ECD.1.1E

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

ASE_ECD.1.2E

The evaluator shall confirm that no extended component can be clearly expressed using existing components.

7.1.5 Stated security requirements (ASE_REQ.1)

7.1.5.1 *Developer action elements:*

ASE_REQ.1.1D

The developer shall provide a statement of security requirements.

ASE_REQ.1.2D

The developer shall provide a security requirements rationale.

7.1.5.2 *Content and presentation elements:*

ASE_REQ.1.1C

The statement of security requirements shall describe the SFRs and the SARs.

ASE_REQ.1.2C

All subjects, objects, operations, security attributes, external entities and other terms that are used in the SFRs and the SARs shall be defined.

ASE_REQ.1.3C

The statement of security requirements shall identify all operations on the security requirements.

ASE_REQ.1.4C

All operations shall be performed correctly.

ASE_REQ.1.5C

Each dependency of the security requirements shall either be satisfied, or the security requirements rationale shall justify the dependency not being satisfied.

ASE_REQ.1.6C

The statement of security requirements shall be internally consistent.

7.1.5.3 *Evaluator action elements:*

ASE_REQ.1.1E

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

7.1.6 TOE summary specification (ASE_TSS.1)

7.1.6.1 *Developer action elements:*

ASE_TSS.1.1D

The developer shall provide a TOE summary specification.

7.1.6.2 *Content and presentation elements:*

ASE_TSS.1.1C

The TOE summary specification shall describe how the TOE meets each SFR.

7.1.6.3 *Evaluator action elements:*

ASE_TSS.1.1E

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

ASE_TSS.1.2E

The evaluator shall confirm that the TOE summary specification is consistent with the TOE overview and the TOE description.

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.

7.5 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 Security Audit, Communication, Cryptographic Support, Identification and Authentication, Security Management, Protection of the TSF, TOE Access, and Trusted Path/Channels. The following table defines which distributed TOE component(s) perform the capabilities described by the SFR.

Table 15: SFR and TOE Component Mapping

Requirement	UEM Server	Android Hub Agent	iOS Hub Agent
[MDMPP] FAU_ALT_EXT.1	X		
[AGENTMOD] FAU_ALT_EXT.2/ANDROID		X	
[AGENTMOD] FAU_ALT_EXT.2/IOS			X
[MDMPP] FAU_GEN.1(1)	X	X	X
[MDMPP] FAU_GEN.1(2)	X		
[AGENTMOD] FAU_GEN.1(2)		X	X
[MDMPP] FAU_NET_EXT.1	X		
[MDMPP] FAU_SAR.1	X		
[AGENTMOD] FAU_SEL.1(2)		X	X
[MDMPP] FAU_STG_EXT.1	X	X	X
[MDMPP] FCO_CPC_EXT.1	X	X	X
[MDMPP] FCS_CKM.1	X	X	X
[MDMPP] FCS_CKM.2	X	X	X
[MDMPP] FCS_CKM_EXT.4	X	X	X
[MDMPP] FCS_COP.1(1)	X	X	X
[MDMPP] FCS_COP.1(2)	X	X	X
[MDMPP] FCS_COP.1(3)	X	X	X
[MDMPP] FCS_COP.1(4)	X	X	X
[MDMPP] FCS_RBG_EXT.1	X	X	X
[MDMPP] FCS_STG_EXT.1	X	X	X
[AGENTMOD] FCS_STG_EXT.1(2)		X	X
[MDMPP] FIA_ENR_EXT.1/ANDROID	X		
[MDMPP] FIA_ENR_EXT.1/IOS	X		
[AGENTMOD] FIA_ENR_EXT.2		X	X
[MDMPP] FIA_UAU.1	X		
[MDMPP] FIA_X509_EXT.1(1)	X	X	X
[MDMPP] FIA_X509_EXT.2	X	X	X
[MDMPP] FIA_X509_EXT.5	X	X	X
[MDMPP] FMT_MOF.1(1)	X		
[MDMPP] FMT_MOF.1(2)	X		
[MDMPP] FMT_MOF.1(3)	X		
[MDMPP] FMT_POL_EXT.1	X		
[AGENTMOD] FMT_POL_EXT.2		X	X
[MDMPP] FMT_SMF.1(1)/ANDROID	X		
[MDMPP] FMT_SMF.1(1)/IOS	X		
[MDMPP] FMT_SMF.1(2)/ANDROID	X		

[MDMPP] FMT_SMF.1(2)/IOS	X		
[MDMPP] FMT_SMF.1(3)	X		
[AGENTMOD] FMT_SMF_EXT.4		X	X
[MDMPP] FMT_SMR.1(1)	X		
[MDMPP] FMT_SMR.1(2)	X		
[AGENTMOD] FMT_UNR_EXT.1		X	X
[MDMPP] FPT_API_EXT.1	X	X	X
[MDMPP] FPT_ITT.1(2)	X	X	X
[MDMPP] FPT_LIB_EXT.1	X	X	X
[MDMPP] FPT_TST_EXT.1	X	[4]	[5]
[MDMPP] FPT_TUD_EXT.1	X	X	X
[MDMPP] FTA_TAB.1	X		
[MDMPP] FTP_ITC_EXT.1	X	X	X
[MDMPP] FTP_ITC.1(1)	X		
[MDMPP] FTP_TRP.1(1)	X		
[MDMPP] FTP_TRP.1(2)	X	X	X

Note: SFRs that originate from the Mobile Device Management Protection Profile are denoted by a [MDMPP], and SFRs that originated from the Mobile Device Management Agent PP-Module are denoted by [AGENTMOD].

The minimum configuration for this evaluation is one UEM Server, and one iOS Hub Agent installed on an Apple device and/or one Android Hub Agent installed on an Android device. Including additional iOS Hub Agents installed on multiple Apple devices and additional Android Hub Agents installed on multiple Android devices as part of an operational configuration will not affect the validity of the functional claims made within the TSS. All TSS descriptions regarding the role, operation, and management of an iOS Hub Agent would be consistent with every additional iOS Hub Agent and Apple device added to the minimum evaluated configuration. All TSS descriptions regarding the role, operation, and management of an Android Hub Agent would be consistent with every additional Android Hub Agent and Android device added to the minimum evaluated configuration. Therefore, all TSS descriptions regarding the iOS Hub Agent and Android Hub Agent can be read with the understanding that the descriptions would apply to one or more of these TOE components and the method in which the additional TOE components met the SFRs would be the same as their minimum configuration equivalent.

Note: The TSS evaluation activities that apply to only the UEM Server component are denoted by [SERVER] and to only the Hub Agent components are denoted by [AGENT]. If the TSS evaluation activity applies to both components, it is not denoted.

8.1 Security Audit

8.1.1 **[MDMPP] FAU_ALT_EXT.1**

[SERVER] The UEM Server component of the TOE provides Authorized Administrators with the ability to view information about enrolled mobile devices and to generate alerts when various events occur. Alerts

[4] TD0438
[5] TD0438

are generated based on configurable “compliance policies” that can detect when a violation has occurred and to mark the affected device as Not Compliant in the device’s overview in the Admin Console.

Authorized Administrators can view information about the status of managed devices through the UEM Admin Console. Two of the dashboards that are accessible from the Main Menu are “Monitor” and “Devices”. From the “Monitor” section of the Admin Console, Authorized Administrators can view the total number of enrolled and unenrolled devices, the total number of compliance violations, devices that failed to install policies (profiles) and which devices have blacklisted apps, devices without required apps, or devices with apps that are not whitelisted. The Authorized Administrator can also view the applications that are associated with particular devices, including application versions. From the “Devices” section of the Admin Console, the Authorized Administrator can view changes in the enrollment status of a device by viewing the enrollment status and enrollment history information. This also lists devices that are enrolled but do not have policies applied to them. The Authorized Administrator can also view the list of compromised (jailbroken/rooted) devices under this section of the Admin Console as well as detailed information about any specific device that the UEM Server knows about.

In addition to being able to review this information on demand, Authorized Administrators can configure the delivery of periodic (daily, weekly, monthly) alert emails from the “Monitor” section of the Admin Console for the following events when they are observed on a device:

- Presence of blacklisted apps
- Presence of non-whitelisted apps
- Absence of required apps
- Compromised (jailbroken or rooted) device
- Last time a device communicated with the UEM Server
- Unapproved model (iOS only)
- Unapproved device manufacturer (Android only)
- Unapproved operating system version (greater than, less than, equal to, not equal to specified version)

The process of rooting an Android device requires the device to be rebooted. During device reboot, the Android Hub Agent will detect the device has been rooted. If the device is connected to Wi-Fi, the Android Hub Agent will send an alert to the UEM Server and will wipe the device. If the device is not connected to Wi-Fi, an alert cannot be sent due to the lack of a connection and the Android Hub Agent will wipe the device. Since performing a device wipe will also remove the Android Hub Agent, this will prevent it from queuing the alert for a rooted device.

8.1.2 **[AGENTMOD] FAU_ALT_EXT.2/ANDROID**

[AGENT] The Android Hub Agent component of the TOE provides the ability to alert the UEM Server in the event that certain behavior on the underlying mobile device is observed. For Android devices, all of the alerting is performed by the Android Hub Agent. The alert for a device becoming enrolled/unenrolled from management is sent by the Android Hub Agent as part of the enrollment/unenrollment process which requires communication with the UEM Server. Alerts for rooted devices occur upon detection by a VMware app after the device reboots and requires Wi-Fi connection to be sent as described under FAU_ALT_EXT.1. All other alerts are based upon policies (profiles) being applied to the mobile device

and the Android Hub Agent collecting information on the device to generate alerts based upon "compliance policies" that detect when a violation has occurred.

The Android Hub Agent is configured by the UEM Server to generate periodic reachability events based upon a configured 'sample interval' and 'transmit interval'. The collecting of information on the device by the Android Hub Agent occurs every 'sample interval'. The Android Hub Agent then queues each sample interval of collected data and will send up to the last 10 sample intervals of collected data to the UEM Server once the 'transmit interval' is reached. If the connection between the Android Hub Agent and UEM Server is down during a 'transmit interval', the Android Hub Agent continues to queue sample intervals of collected data until a connection is available for a 'transmit interval'. The maximum amount of storage is 10 sample intervals. The actual amount of storage for alerts depends on the amount of storage space of the device and the amount the device allocates to the Android Hub Agent app. The alerts generated by the Android Hub Agent based upon the collected data during a 'sample interval' include:

- Successful application of policies
- Generating a periodic reachability event
- Failure to install an application managed by the MAS Server functionality of the UEM Server
- Failure to update an application managed by the MAS Server functionality of the UEM Server
- Detection of blacklisted apps
- Detection of non-whitelisted apps
- Required apps missing
- Unapproved device manufacturer
- Unapproved operating system version

For apps, the UEM Server has the ability to specify if a given application is required, blacklisted, or whitelisted. This assignment can be made both for apps under the control of the UEM Server as well as publicly-available apps on the Google Play Store. If the Android Hub Agent detects a blacklisted app upon the policy being applied to the device (i.e. the app's installation predates the policy), the alerting process immediately remediates the non-compliance by disabling the app. Otherwise, if the policy is applied prior to the installation of the app, the installation of the app is prevented. If the Android Hub Agent detects the absence of required apps as well as the failure to install or update an app managed by UEM Server, the alerting process immediately attempts to remediate the non-compliance by trying to automatically install the app(s) as part of applying the policy requiring those app(s) to the device.

8.1.3 **[AGENTMOD] FAU_ALT_EXT.2/IOS**

[AGENT] The iOS Hub Agent component of the TOE provides the ability to alert the UEM Server in the event that certain behavior on the underlying mobile device is observed. For iOS devices, most of the alerting is performed through the iOS MDM protocol rather than through the iOS Hub Agent. The iOS Hub Agent is responsible for alerting the UEM Server component when a jailbreak is detected and when the device is enrolled/unenrolled from management. Jailbreak detection is performed when the mobile device user launches an app with the VMware jailbreak detection capability, such as the iOS Hub Agent. In the event that a jailbreak is detected and the iOS Hub Agent is unable to communicate with the UEM Server, the notification is continuously retried until communications have been re-established. The alert for a device becoming enrolled/unenrolled from management is sent by the iOS Hub Agent as part of the enrollment/unenrollment process which requires communication with the UEM Server.

The remaining alerts are generated by the underlying iOS platform. These alerts are generated as part of the request and response relationship of an active connection between the iOS platform and the UEM Server; thus, there is no queueing of an alert when a connection is not available. This includes the iOS platform sending alerts when consuming policies (profiles) assigned to it as well as in response to UEM Server querying the iOS platform based upon the UEM Server's periodic reachability event configuration. Additionally, during the periodic reachability events the iOS platform will collect information on the device's model, operating system version, and on installed apps to generate alerts based upon "compliance policies" that detect when a violation has occurred. For apps, the UEM Server has the ability to specify if a given application is required, blacklisted, or whitelisted. This assignment can be made both for apps under the control of the UEM Server as well as publicly-available apps on the Apple Store. iOS does not provide the ability to force a device to install or update apps, thus alerts are only generated if a compliance policy is written to detect the presence of blacklisted apps, non-whitelisted apps, or the absence of required apps. Additionally, the iOS platform will inform the UEM Server when an app fails to install or update on the device that is managed by the MAS Server functionality of the UEM Server.

8.1.4 **[MDMPP] FAU_GEN.1(1)**

The UEM Server, iOS Hub Agent, and Android Hub Agent components of the TOE generate auditable events for their own behavior. These TOE components also rely on their underlying platform to generate audit events. All TOE components rely on their underlying platforms to generate audit logs for their startup and shutdown. The UEM Server generates audit logs for all administrative actions and all commands that are sent to managed devices from the UEM Server. Audit records are generated by the UEM Server for MDM Agent alerts (FAU_ALT_EXT.2). The iOS and Android Hub Agents will also generate audit records defined under [AGENTMOD] FAU_GEN.1(2). The TOE components and the underlying OS platforms also generate audit data for the specific auditable events listed in Table 16 below.

Table 16: Auditable Events by Enforcing Component

Requirement	Auditable Event(s)	Component Generating Record
FAU_ALT_EXT.1	Type of alert.	UEM Server
FCO_CPC_EXT.1	Enabling/Disabling communications between a pair of components.	UEM Server
FCS_CKM.1	Failure of key generation activity for authentication keys.	Windows Platform iOS Platform Android Platform Note: The auditing for this SFR is invoked by the platforms' cryptographic modules.
FCS_RBG_EXT.1	Failure of the randomization process.	Windows Platform iOS Platform Android Platform Note: The auditing for this SFR is invoked by the platforms' cryptographic modules.
FIA_ENR_EXT.1 /ANDROID	Failure of MD user authentication.	UEM Server

Requirement	Auditable Event(s)	Component Generating Record
FIA_ENR_EXT.1/IOS	Failure of MD user authentication.	UEM Server
FIA_X509_EXT.1(1)	Failure to validate X.509 certificate.	Windows Platform iOS Platform Android Hub Agent Android Platform Note: Platform auditing for this SFR for: (1) TLS/HTTPS is invoked by the platforms' mechanisms which implement TLS/HTTPS communication, (2) UEM Server and Hub Agent software signed updates is invoked by the platforms' app software update mechanisms, (3) UEM Server software integrity verification is invoked by Window's Authenticode mechanism, (4) signing profiles is invoked by the Windows platform's signature services, and (5) verifying signed profiles is invoked by the iOS platform's signature services.
FIA_X509_EXT.2	Failure to establish connection to determine revocation status.	Windows Platform iOS Platform Android Hub Agent Android Platform Note: Platform auditing for this SFR for: (1) TLS/HTTPS is invoked by the platforms' mechanisms which implement TLS/HTTPS communication, (2) UEM Server and Hub Agent software signed updates is invoked by the platforms' app software update mechanisms, (3) UEM Server software integrity verification is invoked by Window's Authenticode mechanism, (4) signing profiles is invoked by the Windows platform's signature services, and (5) verifying signed profiles is invoked by the iOS platform's signature services.
FMT_MOF.1(1)	Issuance of command to perform function. Change of policy settings.	UEM Server
FMT_MOF.1(2)	Enrollment by a user.	UEM Server
FMT_SMF.1(2) /ANDROID	Success or failure of function.	UEM Server
FMT_SMF.1(2)/IOS	Success or failure of function.	UEM Server
FPT_ITT.1(2)	Initiation and termination of the trusted channel.	Windows Platform iOS Platform Android Platform Note: Platform auditing for this SFR for TLS/HTTPS is invoked by the platforms' mechanisms which implement TLS/HTTPS communication.
FPT_TST_EXT.1	Initiation of self-test. Failure of self-test.	Windows Platform

Requirement	Auditable Event(s)	Component Generating Record
	Detected integrity violation.	Note: Platform auditing for this SFR for UEM Server software integrity verification is invoked by Window's Authenticode mechanism.
FPT_TUD_EXT.1	Success or failure of signature verification.	Windows Platform iOS Platform Android Platform Note: Platform auditing for this SFR for UEM Server and Hub Agent software signed updates is invoked by the platforms' app software update mechanisms.
FTA_TAB.1	Change in banner setting.	UEM Server
FTP_ITC.1(1)	Initiation and termination of the trusted channel.	Windows Platform Note: Platform auditing for this SFR for TLS/HTTPS is invoked by the Windows platform's mechanism which implements TLS/HTTPS communication.
FTP_TRP.1(1)	Initiation and termination of the trusted channel.	Windows Platform Note: Platform auditing for this SFR for TLS/HTTPS is invoked by the Windows platform's mechanism which implements TLS/HTTPS communication.
FTP_TRP.1(2)	Initiation and termination of the trusted channel.	Windows Platform iOS Platform Android Platform Note: Platform auditing for this SFR for TLS/HTTPS is invoked by the platforms' mechanisms which implement TLS/HTTPS communication.

TOE audit records are recorded with the following format:

> {Syslog Date and Time} {UEM Server IP Address} {Date and Time} {UEM Server Name} AirWatch Syslog Details are as follows Event Type: {EventType}; Event: {Event}; User: {User}; Device Name: {DeviceFriendlyName}; EnrollmentUser: {EnrollmentUser}; Event Source: {EventSource}; Event Module: {EventModule}; Event Category: {EventCategory}; Event Data: {EventData}

The audit records that are generated include at least the following information: date and time of the event {Date and Time}, event type {EventCategory}, subject identity {User}, and success or failure of the event {Event}. When identifying the mobile device this will be in the Device Name: {DeviceFriendlyName} field. Additional contents required by the audit records are usually found in the Event Data: {EventData} field.

The following is an example of an audit record from the UEM Server in order to illustrate the audit record format and the fields contained within these records.

> Oct 2 10:59:05 172.16.72.29 October 02 14:58:16 AirWatch AirWatch Syslog Details are as follows Event Type: Device; ***Event: EnrollmentComplete***; *User: sysadmin*; Device Name: Tester1Local Android Android 9.0 RF8M803LV4L; EnrollmentUser: Tester1Local; Event Source: Server; Event Module:

Enrollment; **Event Category: Enrollment**; Event Data: Url=niap-uem.ssdevrd.com Event Timestamp: October 2, 2019 14:58:15

The audit records that are generated include at least the following information: date and time of the event (underlined text), event type (**bold text**), subject identity (*italicized text*), and success or failure of the event (***bold italicized text***). The type of event and additional audit record contents are described in Table 13. For a full list of audit record examples, refer to Section 8.1.3 of the VMware Workspace ONE Unified Endpoint Management Supplemental Administrative Guidance v1.0.

8.1.5 **[MDMPP] FAU_GEN.1(2)**

[SERVER] The UEM Server component of the TOE, which includes the MAS Server functionality, generates audit records when it is unable to push an application to a managed device or detects that a required application is not present on the device (including failed updates to existing managed applications). This is done by defining a compliance policy that checks for the absence of a specific application which causes the device to generate an audit event on its next periodic check-in if that condition is met. iOS does not provide a mechanism to allow the MAS Server to push an application to the device, but an audit event will be generated if the TSF detects that a required app is not present. For Android, it creates an audit record for each device based upon the compliance policy.

TOE audit records are recorded with the following format:

{Syslog Date and Time} {UEM Server IP Address} {Date and Time} {UEM Server Name} AirWatch Syslog Details are as follows Event Type: {EventType}; Event: {Event}; User: {User}; Device Name: {DeviceFriendlyName}; EnrollmentUser: {EnrollmentUser}; Event Source: {EventSource}; Event Module: {EventModule}; Event Category: {EventCategory}; Event Data: {EventData}

The audit records that are generated include at least the following information: date and time of the event {Date and Time}, event type {EventCategory}, subject identity {User}, and success or failure of the event {Event}. When identifying the mobile device this will be in the Device Name: {DeviceFriendlyName} field. Additional contents required by the audit records are usually found in the Event Data: {EventData} field.

The following is an example of an audit record from the UEM Server in order to illustrate the audit record format and the fields contained within these records.

Nov 20 10:09:46 172.16.72.29 November 20 15:10:00 AirWatch AirWatch Syslog Details are as follows Event Type: Device; Event: InstallApplicationFailed; User: sysadmin; *Device Name: Tester1 Android Android 9.0.0 RF8M31B724F*; EnrollmentUser: Tester1; Event Source: Device; Event Module: CustomDeviceEvents; **Event Category: Command**; Event Data: Event Timestamp: November 20, 2019 15:09:46

The audit records that the MAS Server component creates include the following information: date and time of the event (underlined text), event type (**bold text**), and mobile device identity (*italicized text*). For a full list of audit record examples, refer to Section 8.1.3 of the VMware Workspace ONE Unified Endpoint Management Supplemental Administrative Guidance v1.0.

8.1.6 **[AGENTMOD] FAU_GEN.1(2)**

[AGENT] The iOS and Android Hub Agent component of the TOE and the underlying OS platforms generate audit events for activities on the device. The iOS and Android Hub Agents' underlying platforms

generate audit logs for their startup and shutdown. The TOE components and the underlying OS platforms also generate audit data for the specific auditable events listed in Table 17 below.

Table 17: Agent Auditable Events by Enforcing Component

Requirement	Auditable Event(s)	Component Generating Record
FAU_ALT_EXT.2 /ANDROID	Success/failure of sending alert.	Android Hub Agent
FAU_ALT_EXT.2 /IOS	Success/failure of sending alert.	iOS Platform Note: Platform auditing for this SFR is invoked by the iOS internal MDM agent upon communication with the UEM Server to send alert data.
FAU_SEL.1	All modifications to the audit configuration that occur while the audit collection functions are operating.	iOS Hub Agent iOS Platform Android Hub Agent Note: Platform auditing for this SFR is invoked by the iOS internal MDM agent upon receiving policies from the UEM Server.
FIA_ENR_EXT.2	Enrollment in management.	iOS Platform Android Hub Agent
FMT_POL_EXT.2	Failure of policy validation.	iOS Hub Agent iOS Platform Android Hub Agent Note: Platform auditing for this SFR is invoked by the iOS internal MDM agent upon receiving policies from the UEM Server.
FMT_SMF_EXT.4	Outcome (Success/failure) of function.	iOS Hub Agent iOS Platform Android Hub Agent Android Platform Note: Platform auditing for this SFR is invoked by the iOS internal MDM agent upon receiving policies and commands from the UEM Server.

TOE audit records are recorded with the following format:

> {Syslog Date and Time} {UEM Server IP Address} {Date and Time} {UEM Server Name} AirWatch Syslog Details are as follows Event Type: {EventType}; Event: {Event}; User: {User}; Device Name: {DeviceFriendlyName}; EnrollmentUser: {EnrollmentUser}; Event Source: {EventSource}; Event Module: {EventModule}; Event Category: {EventCategory}; Event Data: {EventData}

The audit records that are generated include at least the following information: date and time of the event {Date and Time}, event type {EventCategory}, subject identity {User}, and success or failure of the event {Event}. When identifying the mobile device this will be in the Device Name: {DeviceFriendlyName} field. Additional contents required by the audit records are usually found in the Event Data: {EventData} field.

The following is an example of an audit record from the iOS Hub Agent in order to illustrate the audit record format and the fields contained within these records.

> Nov 19 11:22:01 172.16.72.29 November 19 16:22:16 AirWatch AirWatch Syslog Details are as follows Event Type: Device; ***Event: CompromisedStatusReported***; *User: sysadmin*; Device Name: Tester1 Android Android 9.0.0 RF8M31B724F; EnrollmentUser: Tester1; Event Source: Device; Event Module: Devices; **Event Category: CompromisedStatus**; Event Data: CompromisedStatus=Compromised;ApplicationVersion=SDK V19.8.0

The audit records that are generated include at least the following information: date and time of the event (underlined text), event type (**bold text**), subject identity (*italicized text*), and success or failure of the event (***bold italicized text***). The type of event and additional audit record contents are described in Table 14. For a full list of audit record examples, refer to Section 8.1.3 of the VMware Workspace ONE Unified Endpoint Management Supplemental Administrative Guidance v1.0.

8.1.7 **[MDMPP] FAU_NET_EXT.1**

[SERVER] Authorized Administrators can use the Admin Console to determine the network connectivity status of devices that have iOS and Android Hub Agents installed on them. Each time a device connects to the UEM Server, the UEM Server will record the time of that connection identifying the device's network connectivity. Devices will connect to UEM Server based upon the iOS and Android Hub Agents' periodic reachability event configuration or in response to an on-demand request by an Authorized Administrator. Android device periodic reachability events are initiated by the Android Hub Agents, iOS device periodic reachability events are initiated by the UEM Server, and the on-demand requests by Authorized Administrators for connectivity status for iOS and Android devices are initiated by the UEM Server.

An Authorized Administrator can make an on-demand request from the Device's dashboard by performing a Device Query. This action will result in the UEM Server determining connectivity status for one or more selected devices by using push notifications. For iOS devices, the TSF uses the Apple Push Notification Service (APNS) to send the request to the device and the device will respond to the request when it has network connectivity. For Android devices, the TSF uses Firebase Cloud Messaging Services (FCM) to send the request to the device and the device will respond to the request when it has network connectivity. Refer to FAU_ALT_EXT.2/IOS and FAU_ALT_EXT.2/ANDROID for more information about how the iOS and Android Hub Agents communicate device statuses back to the UEM Server.

8.1.8 **[MDMPP] FAU_SAR.1**

[SERVER] For audited events that are generated by the UEM Server component of the TOE, the Monitor dashboard on the Admin Console provides administrators with the ability to review audit records. This provides a graphical view of the log data in a human-readable format. Audit data can be searched and sorted using this interface.

For cryptographic behavior that is performed by the UEM Server's underlying platform, auditing is stored in the Windows event logs. These records can also be sorted, filtered, and searched, but this activity is performed using the platform since the TSF is not responsible for generating this data.

Table 16 shows the auditable events that are logged by UEM Server versus its underlying platform. The component used to review the audit data is the same as the component that is used to generate the data to be reviewed.

8.1.9 **[AGENTMOD] FAU_SEL.1(2)**

[AGENT] There is no specific configuration to turn on and off auditing on the TOE, thus the iOS and Android Hub Agents and the underlying OS platforms will always perform auditing. However, an Authorized Administrator creates policies on the UEM Server and will assign them to one or more devices. These policies include requirements for the iOS Hub Agent, iOS Platform, and Android Hub Agent to generate audit records for the functionality configured in the policies. Once the iOS Hub Agent, iOS Platform, and Android Hub Agent receive and apply a policy requiring auditing, they will always generate the necessary audit records. For example, an Authorized Administrator can create a policy that an app is required. The policy will be sent and applied by the Android Hub Agents or iOS platforms on the devices to which the policy has been assigned. If a device does not have the required app, the Android Hub Agent or iOS platform will create an audit record based upon that event type as well as success/failure. Table 20 describes if a policy is 'implemented by' the iOS Hub Agent, iOS Platform, or Android Hub Agent.

The TSF does not support specification of more complex audit pre-selection criteria, such as multiple attributes or logical expressions using attributes.

8.1.10 **[MDMPP] FAU_STG_EXT.1**

[SERVER] In the evaluated configuration, audit data managed by the UEM Server will be transmitted from the UEM Server to a remote Syslog Server over a TLS v1.2 encrypted trusted channel as well as to the SQL database. The actual TLS encryption is handled by the underlying Windows Server 2016 platform. The audit data that are transferred include audit records generated by the UEM Server software as well as audit records that are received from iOS and Android Hub Agents. This does not include audit data that are generated by these TOE components' underlying platforms as this audit data are not managed by the TOE's software boundary. It is therefore the responsibility of the Operational Environment to securely transfer this audit data to a remote location for permanent storage.

The UEM Server is configured to send TOE managed audit data over a specific port to the Syslog Server and if this matches the syslog TLS port, then the Syslog Server will receive a TLS connection initiation. The Syslog Server's certificate is bound to the port, which is defaulted to port 6514 but can be changed by a System Administrator. Once the connection is established, the TOE managed audit logs are sent to the Syslog Server in real time upon the UEM Server creating them or receiving them from Hub Agents. If the Syslog Server connectivity is unavailable, audit records will only be stored in the SQL database while the connection is down. Upon re-establishment of communications with the Syslog Server, new audit records will resume being transmitted to it but the audit records that were generated during the time the Syslog Server connection was down are not sent to the Syslog Server. The SQL database only stores audit records for 30 days before purging them and if maximum capacity for record storage is reached in the SQL database, any new audit records are dropped.

[AGENT] Audit records generated by the iOS and Android Hubs Agents are transmitted to the UEM Server over the HTTPS internal TOE trusted channel. iOS and Android Hubs Agent audit records are generated and sent to the UEM Server in real time. As described above, once these audit records reach the UEM Server, the audit records will be sent to the remote Syslog Server over a TLS v1.2 encrypted trusted channel.

8.2 Communication

8.2.1 **[MDMPP] FCO_CPC_EXT.1**

[SERVER] In the evaluated configuration, the Hub Agents that can join the TOE by enrolling with the UEM Server are limited based upon the allowed DEP identifiers for iOS Hub Agents and the allowed IMEIs and/or serial numbers for Android Hub Agents. The configuration of these enrollment restrictions is the enablement step by the Authorized Administrator through the Admin Console. The entire enrollment process from the UEM Server's perspective is described under FIA_ENR_EXT.1/IOS, FIA_ENR_EXT.1/ANDROID, and FIA_X509_EXT.5. The UEM Server relies on its underlying platform to provide the secure registration channel to the iOS and Android Hub Agents that attempt to enroll and join the TOE. This secure registration channel is described under FTP_TRP.1(2) and is used for all communications between the UEM Server and Hub Agents during the enrollment process. After a Hub Agent has enrolled and joined the TOE, an Authorized Administrator can disable the communication between a Hub Agent and the UEM Server by wiping the Hub Agent's device. This will result in the removal of the Hub Agent and the unenrollment of the device which will prevent communication between the device and the UEM Server.

[AGENT] The entire enrollment process from the Hub Agent's perspective is described under FIA_ENR_EXT.1/IOS, FIA_ENR_EXT.1/ANDROID, FIA_ENR_EXT.2, and FIA_X509_EXT.5. The iOS and Android Hub Agent rely on their underlying platform to provide the secure registration channel to the UEM Server when enrolling into management and joining the TOE. This secure registration channel is described under FTP_TRP.1(2) and is used for all communications between the Hub Agents and UEM Server during the enrollment process.

8.3 Cryptographic Support

[SERVER] Cryptographic services for the UEM Server are provided by the underlying Windows server platform. The Windows Server 2016 platform uses Microsoft's BCryptPrimitives.dll and CNG.sys to perform all cryptographic services.

[AGENT] Cryptographic services for the iOS and Android Hub Agents are mainly provided by the underlying mobile device platforms. The iOS Hub Agent uses Apple iOS platform's CoreCrypto Module to perform all claimed cryptographic services. The Android Hub Agent uses the Android platform's SCrypto and BoringSSL cryptographic modules to perform all claimed cryptographic services, except for the policy digital signature validation requirements. The Android Hub Agent implements OpenSSL for the specific purpose of performing the policy digital signature validation services.

Refer to the platform Security Targets, listed in Section 1.1.5, for more information about the cryptographic functionality provided by the Windows Server, iOS, and Android platforms and their corresponding cryptographic certificates.

8.3.1 **[MDMPP] FCS_CKM.1**

[SERVER] The UEM Server invokes the Windows Server 2016 platform provided functionality for asymmetric key generation in support of TLS communications. The Windows Server 2016 platform provides functionality to support 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 and P-384 that meet FIPS PUB 186-4, “Digital Signature Standard (DSS)”, Appendix B.4.

[AGENT] The iOS and Android Hub Agent’s software invokes the platform provided functionality in support of TLS communications. Both iOS and Android platforms support 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 and P-384 that meet FIPS PUB 186-4, “Digital Signature Standard (DSS)”, Appendix B.4.

8.3.2 **[MDMPP] FCS_CKM.2**

[SERVER] The UEM Server invokes the underlying Window server platform in support of two key establishment schemes for the establishment of TLS communications:

- RSA key establishment conforming to “RSAES-PKCS1-v1_5 as specified in Section 7.2 of RFC 8017” and
- Elliptic curve-based key establishment conforming to NIST Special Publication 800-56A.

[AGENT] The iOS and Android Hub Agent's software relies on the underlying mobile device platform to perform key establishment for TLS communications using the following two key establishment schemes:

- RSA key establishment conforming to “RSAES-PKCS1-v1_5 as specified in Section 7.2 of RFC 8017” and
- Elliptic curve-based key establishment conforming to NIST Special Publication 800-56A

8.3.3 **[MDMPP] FCS_CKM_EXT.4**

[SERVER] The UEM Server invokes the underlying FIPS cryptographic modules to zeroize keys and cryptographic security parameter data when no longer needed. The invoking of key destruction occurs as a result of the UEM Server making a cryptographic function call which requires a key and/or cryptographic security parameter data to be generated by the platform. The platform will therefore perform key destruction when the generated cryptographic data are no longer needed, without requiring a separate function call for key destruction from the UEM Server. All key data maintained by the server platform exists only in volatile memory and are erased by a one-pass overwrite with zeroes followed by a read-verify.

[AGENT] The iOS and Android Hub Agents’ software invokes the underlying mobile device platform to perform key destruction. The invoking of key destruction occurs as a result of the iOS or the Android Hub Agent making a cryptographic function call which requires a key and/or cryptographic security parameter data to be generated by its platform. The platform will therefore perform key destruction when the generated cryptographic data are no longer needed, without requiring a separate function call for key destruction from the iOS or the Android Hub Agent. Key data maintained by the iOS and Android Hub Agents’ platform in volatile memory are erased by a one-pass overwrite with zeroes. Key data maintained by the iOS and Android Hub Agents’ platforms in non-volatile memory are stored in wear-leveled flash memory and are erased by a one-pass overwrite with ones (i.e. block erase).

8.3.4 **[MDMPP] FCS_COP.1(1)**

[SERVER] The UEM Server invokes the underlying Windows Server 2016 platform to perform AES encryption/decryption services for TLS communications and protection of data at rest in platform key storage. All data at rest are protected using AES-GCM-256 as defined in NIST SP 800-38D. Data in transit are protected using either CBC or GCM modes and either 128-bit or 256-bit keys. When operating in CBC mode, the UEM Server's platform conforms to FIPS PUB 197 and NIST SP 800-38A. When operating in GCM mode, the UEM Server's platform conforms to NIST SP 800-38D.

[AGENT] The iOS and Android Hub Agents' software invokes the underlying mobile device platform to perform symmetric encryption/decryption. The iOS and Android Hub Agents' platforms are using either CBC or GCM modes and either 128-bit or 256-bit keys. When operating in CBC mode, the device platforms conform to FIPS PUB 197 and NIST SP 800-38A. When operating in GCM mode, the device platforms conforms to NIST SP 800-38D.

8.3.5 **[MDMPP] FCS_COP.1(2)**

[SERVER] The UEM Server invokes the Window server platform to provide SHA-256, SHA-384 and SHA-512 cryptographic hashing services, conformant to FIPS PUB 180-4, with digest sizes of 256, 384 and 512 bits, respectively. SHA-256 and SHA-384 are used by HMAC in message authentication in TLS communication and SHA-512 is used in the hashing of passwords and for the digital signing of policies.

[AGENT] The iOS and Android Hub Agents' software invokes the underlying mobile device platform to perform cryptographic hashing in support of TLS communication. The iOS and Android Hub Agents' platforms use SHA-256 and SHA-384, conformant to FIPS PUB 180-4, in support of TLS communication.

Additionally, the iOS Hub Agent invokes the underlying mobile device iOS platform for SHA-512 hashing services, conformant to FIPS PUB 180-4, for ECDSA policy digital signature verification.

The Android Hub Agent implements SHA-512 hashing services, conformant to FIPS PUB 180-4, for ECDSA policy digital signature verification. The CAVP SHS certificate number is #C1329.

8.3.6 **[MDMPP] FCS_COP.1(3)**

[SERVER] The UEM Server invokes the Windows Server 2016 platform to provide all digital signature services in accordance with FIPS PUB 186-4. RSA with 2048-bit keys and ECDSA with P-256 and P-384 NIST curves are used for digital signature services in support of TLS communication. Additionally, ECDSA with P-521 NIST curve (using SHA-512) is used for policy signature generation.

[AGENT] The iOS and Android Hub Agents' software invokes the underlying mobile device platform to provide digital signature services in accordance with FIPS PUB 186-4. RSA with 2048-bit keys and ECDSA with P-256 and P-384 NIST curves are used for digital signature services in support of TLS communication.

Additionally, the iOS Hub Agent invokes the underlying mobile device's iOS platform to provide ECDSA with P-521 NIST curve (using SHA-512) services for policy signature verification.

The Android Hub Agent implements the ECDSA with P-521 NIST curve (using SHA-512) services for policy signature verification functionality. The CAVP ECDSA certificate number is #C1329.

8.3.7 [MDMPP] FCS_COP.1(4)

[SERVER] The UEM Server platform invokes the Windows Server 2016 platform to provide keyed-hash message authentication services conformant to FIPS PUBs 180-4 and 198-1. HMAC-SHA-256 and HMAC-SHA-384 are used to perform keyed-hash message authentication, with respective digest sizes of 256 and 384 bits. The key used by HMAC is the UUID which is 160 bits. This key is stored on the Server platform.

[AGENT] The iOS and Android Hub Agents' software invokes the underlying mobile device platform to provide keyed-hash message authentication services conformant to FIPS PUBs 180-4 and 198-1. HMAC-SHA-256 and HMAC-SHA-384 are used to perform keyed-hash message authentication, with respective digest sizes of 256 and 384 bits in support of trusted communication. The key used by HMAC is the UUID which is 160 bits.

8.3.8 [MDMPP] FCS_RBG_EXT.1

Note: The TOE UEM Server and Hub Agent software do not directly invoke their respective platforms' deterministic random bit generator. Instead the TOE's software indirectly invokes their platforms' deterministic random bit generator by directly invoking platform components, which in turn directly invoke the deterministic random bit generator.

[SERVER] The UEM Server invokes the underlying Windows Server 2016 platform to provide random bit generation services. The platform cryptographic module provides an AES counter DRBG (CTR_DRBG) that conforms to NIST SP 800-90A. In order to provide sufficient randomness to the keys generated by this DRBG (as specified in NIST SP 800-57), the DRBG is seeded with at least 256 bits of entropy which is gathered from a platform based RBG.

[AGENT] The iOS and Android Hub Agents' software invokes the underlying mobile device platform to provide random bit generation services. Both iOS and Android platforms implement AES counter DRBG conformant to NIST SP 800-90A. The iOS DRBG is seeded with at least 256 bits of entropy from the platform's software-based noise source. The Android DRBG is seeded with at least 256 bits of entropy from the platform's hardware-based noise source.

8.3.9 [MDMPP] FCS_STG_EXT.1

[SERVER] The TOE platform is responsible for storing keys and relies on its CNG.sys and BCryptPrimitives.dll cryptographic modules to invoke the storage of persistent secrets and private keys which are produced through their operation. All private keys are stored in the Windows Trust Store and all user credentials are stored in authentication repositories. The SQL database Master Key is stored encrypted using an AES-GCM 256-bit key encryption key (KEK). This KEK is encrypted and stored in the Windows Registry. The policy signing X.509v3 certificate is uploaded by the Authorized Administrator and is stored in the SQL database.

The following table contains the list of keys and CSPs for the UEM Server platform:

Table 18: Keys and CSPs for the UEM Server Platform

Key/CSP	Purpose	Origin
AES-CBC	Data in transit	Windows Platform - CNG.sys
AES-GCM	Data at rest	Windows Platform - CNG.sys

	Data in transit	
RSA Public/Private Key	TLS Key Establishment	Windows Platform - CNG.sys
RSA Public/Private Key	Digital Signatures	Windows Platform - CNG.sys
DSA Public/Private Key	KeyGen for DH	Windows Platform - CNG.sys
ECDH Public/Private Key	TLS Key Establishment	Windows Platform - CNG.sys
HMAC Key	Message Authentication	Windows Platform - CNG.sys
RBG CSPs	Random Bit Generation	Windows Platform - CNG.sys and BCryptPrimitives.dll
ECDSA Public/Private Key	KeyGen for ECDH	Windows Platform - CNG.sys
ECDSA Public/Private Key	Digital Signatures	Windows Platform - CNG.sys
X.509v3 Certificates	Digital Signatures	CA Server

[AGENT] The FCS_STG_EXT.1(2) section describes key storage for the iOS and Android Hub Agents.

8.3.10 **[AGENTMOD] FCS_STG_EXT.1(2)**

[AGENT] All iOS Hub Agent keys are stored in the Trust Anchor and all the Android Hub Agent keys are stored in the Android Trust Store of the device. The iOS Hub Agent relies on its platform's CoreCrypto module to invoke the storage of persistent secrets and private keys which are produced through its operation. The Android Hub Agent relies on its platform's BoringSSL and SCrypto to invoke the storage of persistent secrets and private keys which are produced through their operation. These cryptographic modules are invoked by the platform APIs available to the Hub Agents when requesting an encryption function. (see section 8.6.1 for the list of APIs)

The following table contains the list of keys and CSPs for the iOS and Android Hub Agents along with their purpose:

Table 19: Keys and CSPs for the Device

Key/CSP	Purpose	Origin
AES-CBC	Data in transit	iOS Platform - CoreCrypto Android Platform - BoringSSL and SCrypto
AES-GCM	Data in transit	iOS Platform - CoreCrypto Android Platform - BoringSSL and SCrypto
RSA Public/Private Key	TLS Key Establishment	iOS Platform - CoreCrypto Android Platform - SCrypto
ECDH Public/Private Key	TLS Key Establishment	iOS Platform - CoreCrypto Android Platform - BoringSSL
DSA Public/Private Key	KeyGen for DH	iOS Platform - CoreCrypto Android Platform - BoringSSL
HMAC Key	Message Authentication	iOS Platform - CoreCrypto Android Platform - BoringSSL and SCrypto
RBG CSPs	Random Bit Generation	iOS Platform - CoreCrypto Android Platform – BoringSSL and SCrypto
ECDSA Public/Private Key	KeyGen for ECDH	iOS Platform - CoreCrypto

		Android Platform - BoringSSL and SCrypto
X.509v3 Certificates	Digital Signatures	CA Server

8.4 Identification and Authentication

8.4.1 [MDMPP] FIA_ENR_EXT.1/ANDROID

[SERVER] In the evaluated configuration of the TOE, a user enrolls their Android mobile device through a series of steps. First, the user powers on the mobile device and follows the standard Android Setup Assistant instructions, including language, country/region, and Wi-Fi network. Once the device has been set up, the user will need to download the Android Hub Agent from the Google Play Store.

The device user will then enter the UEM Server's IP address into the Android Hub Agent which will be used to establish the enrollment connection that is described under FTP_TRP.1(2). Once the HTTPS/TLS connection between an Android Hub Agent and UEM Server is established, the user provides their credentials to authenticate to the UEM Server and then the enrollment process begins. There are two methods of configuring user authentication for device enrollment:

- **Basic:** The account has a username/password defined by an Authorized Administrator on the UEM Server.
- **LDAP:** The UEM Server is connected to an Active Directory/LDAP Server that is used as a third-party identity store.

The UEM Server can limit the user's enrollment of Android devices based on the device's IMEI and/or serial number, the specific model and/or manufacturer of the device, the number of devices enrolled by a user, and the installed operating system version. An authorized Administrator can configure enrollment restriction policies through the Admin Console based upon these factors. The UEM Server will then initiate the process of having a unique X.509v3 certificate being issued to the Android Hub Agent per the process described under FIA_X509_EXT.5 and will send the MDM profiles (policies) assigned to the device.

In addition, if a device is lost or stolen, a blacklist of devices can be created using serial number/ IMEI/UDID information. Any blacklisting of a device will unenroll it, remove all MDM profiles, and prevent re-enrollment until the device is removed from the blacklist.

8.4.2 [MDMPP] FIA_ENR_EXT.1/IOS

[SERVER] In the evaluated configuration of the TOE, a user enrolls their iOS mobile device through a series of steps. First, an Administrator will enroll the device in Apple DEP which is performed using the device's serial number. Then the user powers on the mobile device and follows the standard iOS Setup Assistant instructions, including language, country/region, and Wi-Fi network. Additionally, the iOS Setup Assistant will continue the enrollment process to the UEM Server through Apple DEP.

As part of enrolling in Apple DEP, the iOS platform will receive the UEM Server's URL which will be used to establish the enrollment connection that is described under FTP_TRP.1(2). Once the HTTPS/TLS connection between an iOS platform and UEM Server is established, the user provides their credentials to authenticate to the UEM Server. There are two methods of configuring user authentication for device enrollment:

- **Basic:** The account has a username/password defined by an Authorized Administrator on the UEM Server.
- **LDAP:** The UEM Server is connected to an Active Directory/LDAP Server that is used as a third-party identity store.

The UEM Server can limit the user's enrollment of specific iOS devices based on device registration with Apple DEP. The DEP registration list of devices' DEP identifiers effectively acts as a device whitelist. This is done by an Authorized Administrator specifying Registered Devices Only in the registration settings; the UEM Server will acquire the list of registered devices through periodic synchronization with Apple DEP. Once authentication is successful, the iOS Hub Agent is then deployed as a managed app by the UEM Server to the iOS mobile device. The UEM Server will then initiate the process of having a unique X.509v3 certificate being issued to the iOS Hub Agent per the process described under FIA_X509_EXT.5 and will send the MDM profiles (policies) assigned to the device.

In addition, if a device is lost or stolen, a blacklist of devices can be created using serial number/ IMEI/UDID information. Any blacklisting of a device will unenroll it, remove all MDM profiles, and prevent re-enrollment until the device is removed from the blacklist.

8.4.3 [AGENTMOD] FIA_ENR_EXT.2

[AGENT] During the enrollment process, the iOS and Android Hub Agents record the UEM Server's DNS name and full URL with hostname. This is the only reference identifier used for the UEM Server. The iOS and Android Hub Agents can only be enrolled with one UEM Server at a time.

8.4.4 [MDMPP] FIA_UAU.1

[SERVER] The UEM Server component of the TOE has a configurable login warning banner which is displayed prior to authentication taking place for both the Admin Console and the Self-Service Portal. There is also a "forgot password" link for the Administrator on the Admin Console that can be used to recover credentials based on username. An email is sent out to the Administrator using the address that is stored by the TOE with a link and once the link is selected, a security question must be answered in order to reset the password. The security questions are supplied over a TLS protected link. The TSF does not actually transmit a password to the Administrator.

In addition, there is an "about" button on the Admin Console homepage that includes the UEM Server software version number, copyright and licensing agreement information.

All other means of interacting with the UEM Server component of the TOE require the Administrator to be authenticated to the Admin Console or the user to be authenticated to the Self-Service Portal.

8.4.5 [MDMPP] FIA_X509_EXT.1(1) and [MDMPP] FIA_X509_EXT.2

[SERVER] The UEM Server relies on the underlying platform to provide X.509v3 certificate services for verification of the code signing of UEM Server software updates, for integrity verification of the UEM Server software, and signing profiles (policies) that are sent to iOS and Android Hub Agents. The X.509v3 certificates used for the UEM Server's software integrity verification and software updates are signed using a public CA certificate during the software build. The UEM Server also relies on its platform to provide its X.509v3 certificate as part of TLS and HTTPS/TLS session establishment in all cases where the UEM Server is the server component in the session (e.g. connections to Hub Agents) as well as upon

the server component's request where the UEM Server is the client component and mutual authentication has been configured. The UEM Server's platform will validate all certificates it receives as part of TLS and HTTPS/TLS session establishment. The HTTPS/TLS connection between the iOS and Android Hub Agents and the UEM Server requires that the administrator bind an X.509v3 certificate to port 443 on the UEM Server. Certificate validity is verified using OCSP. If the UEM Server's TOE platform cannot establish a connection to determine the validity of a certificate, then the certificate is not accepted.

The UEM Server's platform performs the following checks in order to determine if a 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 OCSP as specified in RFC 2560 to verify revocation status.
- 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-1 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.
    - CSP 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.

Additionally, the UEM Server platform's certificate validation service will ensure that all certificate paths terminate with a CA certificate and that all CA certificates include the basicConstraints extension with the CA flag set to TRUE.

[AGENT] The iOS and Android Hub Agents rely on the underlying mobile platform's cryptographic modules to provide X.509v3 certificate services for verification of the code signing of Hub Agent software updates and in support of HTTPS/TLS connections to the Hub Agents. The Hub Agents' software updates are signed using a public CA certificate during the software build. The Hub Agents' platforms will present an X.509v3 certificate as part of the HTTPS/TLS session establishment in all cases where the Hub Agent is the client component and mutual authentication has been configured (e.g. connections to UEM Server).

Additionally, for the iOS Hub Agent only, the platform implements X.509v3 certificate services for the verification of signed profiles (policies) received from the UEM Server that will be applied by the iOS Hub Agent or iOS underlying platform.

Certificate validity is verified using OCSP. If the iOS and Android Hub Agents' platforms cannot establish a connection to determine the validity of a certificate, then the certificate is not accepted.

The iOS and Android Hub Agents' platforms perform the following checks in order to determine if a 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 OCSP as specified in RFC 2560 to verify revocation status.
- 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-1 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.
    - CSP 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.

Additionally, the iOS and Android Hub Agents platforms' certificate validation services will ensure that all certificate paths terminate with a CA certificate and that all CA certificates include the basicConstraints extension with the CA flag set to TRUE, except for certificates related to signed profiles (policies) processed by the Android Hub Agent.

The Android Hub Agent's instance of OpenSSL implements X.509v3 certificate services for the verification of signed profiles (policies) received from the UEM Server. The Android Hub Agent will check certificate validity using OCSP. If the Android Hub Agent cannot establish a connection to determine the validity of a certificate, then the certificate is not accepted.

The Android Hub Agent's instance of OpenSSL performs the following checks in order to determine if a certificate related to signed profiles (policies) 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 OCSP as specified in RFC 2560 to verify revocation status.
- 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-1 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.
    - CSP 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.

Additionally, the instance of OpenSSL on the Android Hub Agent will ensure that all certificate paths terminate with a CA certificate and that all CA certificates include the basicConstraints extension with the CA flag set to TRUE for certificates related to signed profiles (policies).

8.4.6 **[MDMPP] FIA_X509_EXT.5**

[SERVER] The UEM Server platform requires each iOS and Android Hub Agent device to have a unique X.509v3 certificate which is used by UEM Server to perform the client-side authentication of the device as part of the Hub Agent to UEM Server communications that are described under FPT_ITT.1(2). For Android devices, the UEM Server retrieves a specific SCEP challenge for that device from the CA Server. The UEM Server will then bundle the SCEP challenge and the SCEP Server's URL into a payload that is sent to the Android Hub Agent. For iOS devices, the UEM Server will request a unique device certificate over DCOM and include the certificate within the MDM profiles (policies) assigned to the device.

[AGENT] Android devices use SCEP to generate a unique client certificate request. After receiving the payload with the SCEP Server's URL and SCEP challenge, the Android Hub Agent will send the payload to the device. The device will then request its unique X.509v3 certificate from the SCEP Server. Each iOS device receives its unique X.509v3 certificate within the MDM profiles (policies) assigned to the device. Once a Hub Agent receives its unique X.509v3 certificate and the enrollment process is complete, all subsequent communications between a Hub Agent and the UEM Server occurs over the connection described under FPT_ITT.1(2).

8.5 Security Management

8.5.1 **[MDMPP] FMT_MOF.1(1)**

[SERVER] The UEM Server provides the capability to manage its own functionality as well as the behavior of mobile devices that are under management. Authorized Administrators manage the TOE through the UEM Server's Admin Console. The Admin Console allows Administrators to specify the unlock banner for both the Admin Console and Self-Service Portal as well as configure the time period for the periodic checks between the UEM Server and the managed devices, the interaction between TOE components, and the certificate for policy signing (as described in FMT_SMF.1(2)/ANDROID and FMT_SMF.1(2)/IOS below). Finally, the Admin Console provides the ability to configure how audit data are stored and provides the MAS Server capabilities (as described in FMT_SMF.1(3) below).

For the configuration of devices under management, the full list of functions that the TSF can perform and a reference to where in the Admin Console this behavior can be found is described in FMT_SMF.1(1)/ANDROID and FMT_SMF.1(1)/IOS below. Note that iOS does not provide the ability for third-party MDM software to configure certain aspects of the device's functionality.

As discussed in FMT_SMR.1(1) below, the TOE provides the ability to define multiple administrative roles, each with its own set of authorized permissions. Individual accounts can be assigned these administrative roles and can also be scoped to only have the authority to manage certain devices or collections of devices based on group membership. If an administrator belongs to a role that has the privilege to perform a certain action and the target for that action is within the scope of their group membership, they are considered to be an Authorized Administrator for the requested function for the purposes of this SFR.

8.5.2 **[MDMPP] FMT_MOF.1(2)**

[SERVER] The UEM Server's Authorized Administrator configures the enrollment process. For its own functionality, the Admin Console provides the ability to configure the certificates that are used by the UEM Server as well as the devices that are permitted to enroll in management. This option also allows a maximum number of devices per user to be configured for enrollment.

For both iOS and Android devices, enrollment is performed by a MD user or Authorized Administrator with physical custody of the mobile device. Note that in order to enroll any device, valid user credentials are required which are verified by the UEM Server and/or the external Active Directory/LDAP Server. Since Authorized Administrators are responsible for the creation of MD user accounts on UEM Server, they are able to perform first-use actions requiring user authentication prior to the MD user accessing the account and changing their password.

Note that for iOS, enrollment of mobile devices is brokered using Apple DEP. In the evaluated configuration, the UEM Server is configured to specify the use of registered devices only. Authorized Administrators ensure that iOS mobile devices are first registered with DEP so that they can be selected for enrollment.

8.5.3 **[MDMPP] FMT_MOF.1(3)**

[SERVER] The UEM Server provides two methods of restricting the download of apps: through "smart groups" and through the use of whitelisting and blacklisting.

The UEM Server uses "smart groups" to separate devices based on how policies are applied to them. Smart groups can consist of organization groups, user groups, and device characteristics. If a device belongs to the smart group, whether it was individually assigned, has characteristics (such as operating system version or model type) associated with the smart group, or is owned by a user who belongs to the group, an app may be configured to be made available to download on demand from the MAS Server functionality of UEM Server. Authorized Administrators have the ability to construct smart groups from these other groups. For any applications that reside in the UEM Server's MAS Server functionality or public applications that are referenced through external links, the Authorized Administrator has the ability to assign one or more smart groups to the app to push it to a set of devices or make it available to be downloaded by them. This assignment can be used to determine if the app is automatically pushed to certain devices based on smart group membership or if it is available on demand.

Apps can also be assigned to application groups. These groups are used to collect apps into a bundle of required apps, blacklisted apps, or whitelisted apps. The application group is then associated with one or more user and/or organizational groups in order to specify what devices to which it applies. This categorization of apps and subsequent association with managed devices allows the iOS and Android Hub Agents to alert the UEM Server if a managed device has one or more apps that are in violation of policy.

Note that a whitelist and blacklist policy can both be applied to the same device. In this case, the app whitelist acts as an exception to apps on the blacklist so they can be installed. This occurs when a device is part of multiple smart groups.

There are differences in how policies for required apps, blacklisted apps, and whitelisted apps are enforced on Android and iOS devices based upon the functionality provided to an MDM through the operating system's APIs:

- For Android devices, a blacklisted app cannot be installed after the policy with the blacklist is applied to the device. If the app is already installed when the blacklist policy is applied, the Android Hub Agent will disable the app which makes it non-executable. The disable functionality is performed instead of removal of the app for BYOD purposes. If a single app is whitelisted, all other apps are considered blacklisted (except system apps).
- For iOS devices, iOS does not provide the ability to automatically push apps onto a device or the mandatory prohibition of blacklisted or non-whitelisted apps, so all enforcement of required/whitelisted/blacklisted apps is handled in a reactive manner. Additionally, if an iOS app is specified as automatically pushed to the device, the MD user will still be prompted to accept the app before it is downloaded and installed.

8.5.4 **[MDMPP] FMT_POL_EXT.1**

[SERVER] The UEM Server provides policies (known as "profiles") that can be assigned to groups of devices. Profiles are transmitted to the assigned devices and are applied to the device by one of the following entities: the iOS Hub Agent, the iOS underlying platform, or the Android Hub Agent. For iOS devices, the entity applying the policy depends on the type of policy being sent. The UEM Server will digitally sign the policies with an X.509v3 certificate and the signature will be validated by the entity receiving the policy before to the entity applies the policy to the device. All policies are signed by the UEM Server with a trusted CA certificate using ECDSA with SHA-512. Since the devices have a finite set of trusted root certificates that they permit, the Authorized Administrator is expected to acquire certificates from a trusted third-party (e.g., GoDaddy, VeriSign) for signing the policies.

8.5.5 **[AGENTMOD] FMT_POL_EXT.2**

[AGENT] Candidate profiles are sent to the device by the UEM Server when they are assigned to that device. These profiles have been signed by the UEM Server and require verification before they are applied on the device. For Android devices, all policies are applied by the Android Hub Agent and are signed with a trusted CA certificate using ECDSA with SHA-512. For iOS devices, profiles that are applied by the iOS Hub Agent and iOS underlying platform are signed with a trusted CA certificate using ECDSA with SHA-512. The Android Hub Agent will use its instance of OpenSSL for the verification of signed policies for which it will apply to the device. The iOS Hub Agent and iOS underlying platform will use the iOS underlying platform for the verification of signed policies for which they will apply to the device. Verification of a policy's signature is performed as described under the FIA_X509_EXT.1(1) and FIA_X509_EXT.2 section. The iOS Hub Agent, the iOS underlying platform, and the Android Hub Agent will only apply a policy that has been signed by the UEM Server's certificate and that certificate is determined to be valid. If a policy is received that is not signed, signed by an incorrect certificate, or the certificate is deemed invalid, the policy will not be applied on the device.

8.5.6 **[MDMPP] FMT_SMF.1(1)/ANDROID and [MDMPP] FMT_SMF.1(1)/IOS**

[SERVER] The UEM Server component of the TOE has the ability to issue commands and configuration policies to mobile devices. Depending on the mobile device platform and the function being configured, these may be transmitted to the device itself through the platform's capabilities or to the iOS and Android Hub Agent component of the TOE that resides on the device. This is dependent on what APIs the device operating system makes available to third-party applications to access remotely.

The following table lists the management functions that can be performed by the UEM Server as defined by the MDM PP, how those functions are initiated, as well as whether this behavior is enforced by the iOS and Android Hub Agent or by the underlying mobile device platform. Unless specified otherwise, the management function is initiated from the device Details View in the Admin Console.

Table 20: UEM Server Management Functions

iOS			Android		
Command	**Claimed in VID10937[6]**	**Implemented By**	**Command**	**Claimed in VID10979[7]**	**Implemented By**
1. transition to the locked state – "Lock" button.	Yes	Platform	**1. transition to the locked state** – "Lock" button.	Yes	Hub Agent
2. full wipe of protected data – "More" Actions" button > Device Wipe.	Yes	Platform	**2. full wipe of protected data –** "More Actions" button > Device Wipe.	Yes	Hub Agent
3. unenroll from management – "More" Actions" button > Device Wipe.	Yes	Platform	**3. unenroll from management** – "More Actions" button > Enterprise Wipe.	No	Hub Agent
4. install policies – assigned and applied to target devices at the creation or modification of a profile under Devices > Profiles & Resources > Profiles.	No	Platform	**4. install policies** – assigned and applied to target devices at the creation or modification of a profile under Devices > Profiles & Resources > Profiles.	No	Hub Agent
5. query connectivity status – "Query" button.	No	Platform	**5. query connectivity status** – "Query" button.	No	Hub Agent
6. query the current version of the MD firmware/software – "Query" button. Status shown in the main detail view page.	No	Platform	**6. query the current version of the MD firmware/software** – "Query" button. Status shown in the main detail view page.	No	Hub Agent
7. query the current version of the hardware model of the device – "Query" button. Status shown in the main detail view page.	No	Platform	**7. query the current version of the hardware model of the device** – "Query" button. Status shown in the main detail view page.	No	Hub Agent
8. query the current version of installed mobile applications – "Query" button. Status shown in the Apps tab under the main detail view page.	No	Platform	**8. query the current version of installed mobile applications** – "Query" button. Status shown in the Apps tab under the main detail view page.	No	Hub Agent
9. import X.509v3 certificates into the Trust Anchor Database – assigned and applied to devices as part of a policy under the "Credentials" tab when defining the policy.	Yes	Platform	**9. import X.509v3 certificates into the Trust Anchor Database** – assigned and applied to devices as part of a policy under the "Credentials" tab when defining the policy.	Yes	Hub Agent

[6] TD0479
[7] TD0479

10. install applications – Apps and Books tab, Details View. Admin will be prompted to define what devices an application is assigned to during definition or modification of the application. When the application is specified as automatic distribution, the installation is initiated by the TSF.	Yes	Platform	**10. install applications –** Apps and Books tab, Details View. Admin will be prompted to define what devices an application is assigned to during definition or modification of the application. When the application is specified as automatic distribution, the installation is initiated by the TSF.	Yes	Hub Agent
11. update system software – the UEM Server will send command to the iOS platform to update to the latest OS, then the iOS platform will reach out to Apple to get the latest OS.	Yes	Platform	**11. update system software –** the UEM Server will send command to Samsung's Enterprise Firmware Over the Air (E-FOTA) Server to update to the latest OS, then the E-FOTA Server will push the updated software to the devices.	Yes	Platform
12. remove applications – only Enterprise applications can be removed on iOS, refer to Function 13	Yes	Platform	**12. remove applications –** only Enterprise applications can be removed on Android, refer to Function 13	Yes	Hub Agent
13. remove Enterprise applications – specific application from a single device: Device details, Apps tab, Remove option ("X") button for the desired application.	Yes	Platform	**13. remove Enterprise applications –** specific application from a single device: Device details, Apps tab, Remove option button for the desired application.	Yes	Hub Agent
14. wipe Enterprise data – "More Actions" button > Enterprise Wipe.	Yes	Platform	**14. wipe Enterprise data –** "More Actions" button > Enterprise Wipe.	Yes	Hub Agent
15. remove imported X.509v3 certificates – "More" tab > "Certificates", Revoke option.	Yes	Platform	**15. remove imported X.509v3 certificates –** "Devices" > Profiles > choose the profile that pushed the certificate > "Devices" > "Remove Profile"	Yes	Hub Agent
16. alert the user – "Send" button. Note that this refers to alerting the user of the mobile device, not an Administrator on the UEM Server. This can be sent as an email, SMS, or push notification.	No	Hub Agent (push notification), Platform (SMS)	**16. alert the user –** "Send" button. Note that this refers to alerting the user of the mobile device, not an Administrator on the UEM Server. This can be sent as an email, SMS, or push notification.	No	Hub Agent (push notification), Platform (SMS)
22. place applications into application process groups – Apps & Books > Applications	No	Platform	**22. place applications into application process groups –** Apps & Books >	No	Hub Agent

> Applications Settings > App Groups.			Applications > Applications Settings > App Groups.		
23. revoke Biometric template – by deleting the passcode, this disables the biometric template for use.	No	Platform			
25. password policy – defined in the Passcode properties of a profile.	Yes	Platform	**25. password policy –** defined in the Passcode properties of a profile.	Yes	Hub Agent
26. session locking policy – Defined in the Passcode properties of a profile.	Yes	Platform	**26. session locking policy –** Defined in the Passcode properties of a profile.	Yes	Hub Agent
27. wireless networks (SSIDs) to which the MD may connect – Defined under the Wi-Fi properties of a profile.	No	Platform	**27. wireless networks (SSIDs) to which the MD may connect –** Defined under the Wi-Fi properties of a profile.	Yes	Hub Agent
28. security policy for each wireless network – defined in the Wi-Fi properties of a profile; the permitted CA(s) are defined by reference on the Wi-Fi properties to those defined under Credentials.	Yes	Platform	**28. security policy for each wireless network –** defined in the Wi-Fi properties of a profile; the permitted CA(s) are defined by reference on the Wi-Fi properties to those defined under Credentials.	Yes	Hub Agent
29. application installation policy – groups of required, whitelisted, and/or blacklisted apps can be defined in Apps & Books > Application Settings >App Groups. Note that iOS does not provide a mechanism to pre-emptively enforce application whitelisting/blacklisting but the TOE can take corrective action if a compliance policy is defined to detect the presence of a blacklisted or non-whitelisted app.	Yes	Platform	**29. application installation policy –** groups of required, whitelisted, and/or blacklisted apps can be defined in Apps & Books > Application Settings >App Groups.	Yes	Hub Agent
30. enable/disable policy for camera and screen capture across device – defined in the Restrictions properties of a profile.	Yes	Platform	**30. enable/disable policy for camera, microphone, and screen capture across device –** defined in the Restrictions properties of a profile.	Yes	Hub Agent
31. enable/disable policy for the VPN across MD and on a per-app basis – defined in the VPN properties of a profile or in the "VPN Access" setting for an individual app assignment.	Yes	Platform	**31. enable/disable policy for the VPN across MD –** defined in the VPN properties of a profile.	Yes	Hub Agent
			32. enable/disable policy for Wi-Fi, cellular, Bluetooth, and NFC – defined in the	Yes	Hub Agent

			"Restrictions" tab of a profile.		
			33. enable/disable policy for data signaling – defined in the "Restrictions" tab of a profile.	No	Hub Agent
			34. enable/disable policy for Wi-Fi tethering, USB tethering, and Bluetooth tethering – defined in the "Restrictions" tab of a profile.	Yes	Hub Agent
			35. enable/disable policy for developer modes – defined in the "Restrictions" tab of a profile.	Yes	Hub Agent
36. enable policy for data-at-rest protection – For iOS devices, data-at-rest protection is automatically enabled if a passcode is set so this is configured under the Passcode properties of a profile.	Yes	Platform	**36. enable policy for data-at-rest protection –** "Require Storage Encryption" and "Require SD Card Encryption" options in the Passcode tab of a profile.	Yes	Hub Agent
			37. enable policy for removable media's data-at-rest protection – "Require SD Card Encryption" from the Passcode tab of a profile.	Yes	Hub Agent
40. enable/disable policy for display notification in the locked state – can enable/disable any notification on a per-app basis based upon the bundle ID.	Yes	Platform	**40. enable/disable policy for display notification in the locked state** – can enable/disable all notification through "Allow Notifications" defined in the "Restrictions" properties of a profile.	Yes	Hub Agent
47. the unlock banner policy – configured through the 'if lost return' function.	Yes	Platform	**47. the unlock banner policy** – "Lockscreen Overlay" under the Passcode tab in a profile.	Yes	Hub Agent
			49. enable/disable USB mass storage mode – defined in the "Restrictions" properties of a profile.	Yes	Hub Agent
50. enable/disable backup – defined in the "Restrictions" tab of a profile under the iCloud subcategory.	No	Platform	**50. enable/disable backup –** defined in the "Restrictions" tab of a profile.	No	Hub Agent
			52. enable/disable location services – defined in the "Restrictions" tab of a profile.	Yes	Hub Agent
			53. enable/disable policy for user unenrollment – defined	No	Hub Agent

			in the "Restrictions" tab of a profile.		
			54. enable/disable policy for the Always-On VPN protection across device – defined in the VPN properties of a profile.	Yes	Hub Agent
55. enable/disable policy for use of Biometric Authentication Factor – defined in the "Restrictions" tab of a profile.	Yes	Platform	**55. enable/disable policy for use of Biometric Authentication Factor** – defined in the Passcode properties of a profile.	Yes	Hub Agent
			58. enable/disable automatic updates of system software – defined on the Restrictions properties of a profile, "Allow OTA Upgrade"	No	Hub Agent
			59. enable/disable removable media – defined on the Restrictions properties of a profile, "Allow SD Card Access"	No	Hub Agent
60. application installation policy – groups of required, whitelisted, and/or blacklisted apps can be defined in Apps & Books > Application Settings >App Groups. Note that iOS does not provide a mechanism to pre-emptively enforce application whitelisting/blacklisting but the TOE can take corrective action if a compliance policy is defined to detect the presence of a blacklisted or non-whitelisted app. Note: Command #60 is an extension of Command #29 since allowed applications are only managed by name and not version.	Yes	Platform	**60. application installation policy –** groups of required, whitelisted, and/or blacklisted apps can be defined in Apps & Books > Application Settings >App Groups. Note: Command #60 is an extension of Command #29 since allowed applications are only managed by name and not version.	Yes	Hub Agent
61. iOS Hub Agent passcode authentication policy – specifying complexity requirements for authenticating to the Hub Agent can be defined in Settings > Apps > Settings and Policies > Security Policies	No	Hub Agent			

8.5.7 [MDMPP] FMT_SMF.1(2)/ANDROID and [MDMPP] FMT_SMF.1(2)/IOS

[SERVER] The UEM Server provides the ability to manage its own behavior. Listed below are the internal management functions that are provided along with information about how those functions are performed:

- **Configuration of X.509v3 certificates for UEM Server use**: UEM Server utilizes the X.509v3 certificate which is imported into the "Personal" certificate category for Windows Server 2016 Computer account on which UEM Server is installed.
- **Configure devices permitted for enrollment based on:**
  - **Android Devices: devices specified by IMEI or serial number, specific device models, number of devices, manufacturer, and operating system**: Defined in Groups & Settings > All Settings > Devices & Users > General > Enrollment and choosing the Restrictions selection. The Restrictions tab allows for customize enrollment restriction policies.
  - **iOS Devices: devices specified by DEP identifier**: Defined in Devices & Users > General > Enrollment where Current Setting is set to Override and Devices Enrollment Mode is set to Registered Devices Only.
- **Configuration of TOE unlock banner**: Defined for the Admin Console in Settings under System > Branding.
- **Periodicity of agent communications to query connectivity status, query current version of the MD firmware/software, query the current version of the device hardware model, query the current version of installed mobile apps**:
  - Android Devices: Set globally for the supported Android device platform as follows: Groups and Settings > All Settings > Devices and Users > Android > Intelligent Hub Settings
  - iOS Devices: Set globally for the supported iOS device platform as follows: Groups and Settings > All Settings > Devices and Users > Apple > MDM Sample Schedule.
- **Configure the privacy-sensitive information that will and will not be collected from particular mobile devices**: Located in the Admin Console under Groups & Settings > All Settings > Devices & Users > General > Privacy. The types of data categories collected are GPS, Telecom, Applications, Profiles, and Network.
- **Configure the interaction between TOE components:**
  - Android Devices: allow devices based upon IMEI or serial number: Defined in Groups & Settings > All Settings > Devices and Users > General > Enrollment
  - iOS Devices: allow devices based upon DEP identifier: Defined in Devices & Users > General > Enrollment where Current Setting is set to Override and Devices Enrollment Mode is set to Registered Devices Only.
- **Configure server administrator login session timeout:** Set through the Admin Console under Groups & Settings > All Settings > Admin > Console Security > Session Management
- **Configure Enterprise certificate to be used for signing policies:** Set through the Admin Console under Groups & Settings > All Settings > System > Advanced > Policy Signing Certificate
- **Configure MDM Agent/platform to perform a network reachability test:**
  - Android Devices: Set through the Admin Console under Groups & Settings > All Settings > Devices & Users > Android > Intelligent Hub Settings

  - iOS Devices: Set through the Admin Console under Groups & Settings > All Settings > Devices & Users > Apple > MDM Sample Schedule
- **Configure transfer of MDM server logs to another server for storage, analysis, and reporting:** Syslog can be configured either under Groups & Settings > All Settings > System > Enterprise Integration > Syslog. Or by navigating to: Monitor > Reports & Analytics > Events > Syslog (they point to the same location).

8.5.8 [MDMPP] FMT_SMF.1(3)

[SERVER] The MAS Server component of the UEM Server provides the ability to configure application access in the Admin Console. The MAS Server is defined in the Admin Console under Apps & Books > Applications. Applications can be added to the MAS Server, either as an individual file that is uploaded to the server itself, or as a URL reference to an externally-stored application such as one that resides within the Apple App Store and the Google Play Store. When an application is defined in the MAS Server, it is assigned to one or more smart groups, which are defined under Groups and Settings in the Admin Console. Smart groups can consist of one or more organization groups, user groups, or devices that share certain characteristics regardless of owner. These assignments can be used to define if the application is automatically downloaded onto the impacted devices or is simply made available by the MAS Server to be downloaded on demand by the user.

The MAS Server also provides the ability to configure application access groups so that different applications can be flagged as required, whitelisted, or blacklisted. These application groups are assigned a type (required, whitelisted, blacklisted) and associated with a smart group in the same way that individual applications are assigned. The UEM Server can then define compliance policies to generate alerts when required groups are missing or when prohibited/non-whitelisted apps are present.

8.5.9 [AGENTMOD] FMT_SMF_EXT.4

[AGENT] The iOS and Android Hub Agents have the ability to interact with the underlying mobile device platform in order to enforce the UEM Server management functions. All commands and configuration policies that are defined in FMT_SMF.1(1)/IOS and FMT_SMF.1(1)/ANDROID that are received by the iOS and Android Hub Agents respectively, result in the mobile device platform being queried or modified in some way. This also includes the ability to upload certificates into the device's certificate store that are used to establish trusted communications between the iOS and Android Hub Agents and the UEM Server as part of the enrollment of the device as described under FIA_X509_EXT.5. Information about how the iOS and Android Hub Agents may enforce these management functions depends on the mobile device platform and is described under FMT_POL_EXT.2. The iOS and Android Agents can also prevent the users from unenrolling from management as described under FMT_UNR_EXT.1.

The Android Hub Agent is configured by the UEM Server to generate periodic reachability events based upon a configured 'sample interval' and 'transmit interval'. The collecting of information on the device by the Android Hub Agent occurs every 'sample interval'. The Android Hub Agent then queues each sample interval of collected data, and will send up to the last 10 sample intervals of collected data to the UEM Server once the 'transmit interval' is reached.

The iOS Hub Agent also has the ability to configure the periodicity of reachability events by enforcing the sampling interval values that are configured on the UEM Server. When the UEM Server communicates

with the iOS Hub Agent for policy/sample data, this is considered to be a reachability event since the outcome of this activity updates the Last Seen time of the device in the Admin Console. iOS Hub Agent is based upon request/response with the UEM Server, so there is no way for events to be generated when a connection is not made to the UEM Server.

8.5.10 **[MDMPP] FMT_SMR.1(1)**

[SERVER] Roles and permissions are configurable via the Admin Console. Permissions enable and disable specific access to features within the Admin Console and determines if read/write or read-only access is granted to these features. The Admin Console also defines admin groups based on Active Directory/LDAP group information so that individual Administrator accounts can be scoped to only interact with users and/or devices that match their own group membership. When an Administrator is attempting to perform a management function on the TOE, they will not be considered an "Authorized Administrator" unless both of the following are true:

- They are performing an action that is allowed based on the permissions granted to their assigned admin role.
- They are accessing an object that is within the scope of their admin group membership. If the Administrator has no assigned admin group, all objects are within their authorized scope.

While there are several default admin roles, the Admin Console provides the ability for additional admin roles to be created, each with their own set of allowed privileges. Admin group assignment is done at the account level rather than the role level, so two accounts can be assigned the same admin role but belong to different groups. In the evaluated configuration, the TSF will include the following roles:

- **Server primary administrator**: responsible for server installation, initial configuration, and maintenance functions. Responsible for the setup and maintenance of security configuration administrator and auditor accounts.
- **Security configuration administrator**: responsible for security configuration of the server, setup and maintenance of mobile device profiles, definition of user groups, and setup and maintenance of the device user group administrator role, its members, and its permissions.
- **Device user group administrator**: responsible for maintenance of user accounts, including setup, change of account configurations, and account deletion.
- **Auditor**: responsible for review and maintenance of server and device audit logs.

The specific permissions associated with each of these roles are defined in the supplemental administrative guidance for the TOE.

An administrator account may only be assigned one admin role at a time. The "administrator" role as defined by FMT_SMR.1.1(1) is intended to encompass any of the individual roles listed above.

Users (or "MD users") are defined separately from administrators. If someone is defined as a user in the Admin Console, they will not be defined as an administrator. A user is simply an individual who possesses a mobile device that is enrolled in management. A user is able to access the Self-Service Portal but not the Admin Console. However, a user can have multiple roles associated with their account but can only be logged into one role at a time. When the user logs in, they can change their role and do not have log out of the system and re-authenticate to log back into the system.

8.5.11 **[MDMPP] FMT_SMR.1(2)**

[SERVER] The MAS Server is logically integrated with the UEM Server. It is accessed by Administrators using the Apps & Books tab in the Admin Console. Since this is not accessed separately from the remainder of the UEM Server capabilities, the administrative roles that can interact with the MAS Server are defined in the same manner as for FMT_SMR.1(1) above. The UEM Server also maintains the roles of enrolled mobile devices and application access groups.

8.5.12 **[AGENTMOD] FMT_UNR_EXT.1**

[AGENT] In the evaluated configuration, the "Block User Unenrollment" will be configured for all Android devices which will prevent the unauthorized removal of the Android Hub Agent's software from the mobile device. When configured in this manner, the Android Hub Agent will perform the following actions to prevent unenrollment:

- Removes the unenrollment button;
- Disables the user from demoting the Android Hub Agent from a device Administrator (preventing uninstall); and
- Removes the ability to uninstall the Android Hub Agent through the Google Play Store.

For the iOS Hub Agent, Apple DEP provides the unenrollment protection mechanism for the TOE through the use of the Lock MDM Profile feature. The iOS Hub Agent leverages the functionality provided by the underlying device platform, which has been enrolled in Apple DEP, to prevent the unauthorized removal of the iOS Hub Agent software.

8.6 Protection of the TSF

8.6.1 **[MDMPP] FPT_API_EXT.1**

[SERVER] The UEM Server uses only the supported Windows Server 2016 platform APIs listed below in order to function.

- .NET API
- Diagnostic API (Windows event logs)
- Networking and Internet API (Microsoft Internet Information Services (IIS))
- Security and Identity API (Windows Authentication)

[AGENT] When installed on a mobile device with the Android OS, the TOE uses only the supported platform APIs listed below in order to function.

- java.security.KeyStore
- Javax.net.HttpURLConnection
- Javax.net.ssl
- KeyChain API (Android Platform Keystore)
- KeyMaster API
- SCrypto
- BoringSSL

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

- Common Crypto
- CoreCrypto
- Security.framework

8.6.2 **[MDMPP] FPT_ITT.1(2)**

[SERVER] The UEM Server's platform uses HTTPS/TLS (TLS v1.2) to provide a trusted communication between the UEM Server and enrolled iOS and Android Hub Agents. These protocols are used to protect the data traversing the channel from disclosure and/or modification. This internal channel is used between the UEM Server and the iOS and Android Hub Agents for all communications between these TOE components after device enrollment. Since the MAS Server is logically integrated with the UEM Server, the trusted channel to secure its communication with the iOS and Android Hub Agents is the same. The UEM Server's platform identity is validated through its X.509v3 certificate presented during TLS session establishment. The UEM Server's platform is invoked by the Hub Agent platforms' making a HTTPS/TLS connection request. During TLS session establishment, the UEM Server's platform will also validate the iOS or Android platform's presented X.509v3 certificate to validate their identities. In the evaluated configuration, the UEM Server's platform will be configured to allow only the following ciphersuites:

- TLS_RSA_WITH_AES_128_CBC_SHA256 as defined in RFC 5246
- TLS_RSA_WITH_AES_256_CBC_SHA256 as defined in RFC 5246
- TLS_ECDHE_ECDSA_WITH_AES_128_CBC_SHA256 as defined in RFC 5289
- TLS_ECDHE_ECDSA_WITH_AES_256_CBC_SHA384 as defined in RFC 5289
- 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_256_GCM_SHA384 as defined in RFC 5289

[AGENT] The iOS and Android Hub Agents rely on their underlying platforms to provide the HTTPS/TLS communication path and the validate the UEM Server's X.509v3 certificate. The iOS and Android Hub Agents' identities are validated through their X.509v3 certificate presented during TLS session establishment. The iOS and Android Hub Agents' platforms always initiate this internal channel based upon the iOS platform or Android Hub Agent receiving a reachability request to the UEM Server, or an event occurs requiring the iOS Hub Agent (e.g. jailbreak detection) or Android Hub Agent (e.g. transmit internal) to initiate a connection.

8.6.3 **[MDMPP] FPT_LIB_EXT.1**

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

[SERVER] The UEM Server uses only the listed third-party dynamic libraries for Windows Server 2016 in Appendix A, Section 9.1, in order to function.

[AGENT] When installed on a mobile device with the Android OS, the TOE uses only the listed third-party dynamic libraries in Appendix A, Section 9.2, in order to function.

When installed on a mobile device with iOS, the TOE uses only the listed third-party libraries in Appendix A, Section 9.3, in order to function.

8.6.4 [MDMPP] FPT_TST_EXT.1

[SERVER] The UEM Server .dll files and executable code are digitally signed using an X.509v3 certificate from a public CA certificate. During initial installation of the UEM Server and each time the server application is started, the native Windows Authenticode process is invoked to validate the integrity of the UEM Server. If the validation fails, the native Windows Authenticode process will terminate the host process and the service will not start.

In addition, the UEM Server uses the FIPS 140-2 validated cryptographic modules that belong to the underlying Windows Server 2016 platform. These modules each perform their own power-up self-tests upon initial start-up, including cryptographic algorithm known answer tests (KATs) and an integrity verification check.

These tests are sufficient to validate the correct operation of the TSF because they verify that the platform's cryptographic modules which the UEM Server relies upon are operating correctly, the platform does an integrity check of the UEM Server's software, and that the underlying hardware does not have any anomalies that would cause the software to be executed in an unpredictable or inconsistent manner.

8.6.5 [MDMPP] FPT_TUD_EXT.1

[SERVER] Updates for the UEM Server are downloaded as a zip package from the VMware support website. An Authorized Administrator can login to https://support.workspaceone.com/, then navigate to Software > Console, or at https://resources.workspaceone.com/software/console. The UEM Server software updates are installed by the underlying platform directly onto the system; the platform does not have an automatic method of pulling down or installing the updates without Authorized Administrator (local authorized administrator of the platform) initiation via the platform. The updates are digitally signed using a Digicert X.509v3 certificate which is installed in the Windows trusted key store on the underlying platform which verifies the software updates. The before and after version numbers of the UEM Server software can be checked by clicking on the "About" button on the UEM Server's Admin Console. Each iOS and Android Hub Agent's current software version can also be queried through the UEM Server's Admin Console by an Authorized Administrator.

[AGENT] Updates to the Hub Agents' software are provided by the Google Play Store (Android), Apple Store (iOS), or the UEM Server store over HTTPS/TLS. The Hub Agents' software updates are signed using a public CA certificate during the software build and loaded onto 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. The software update is downloaded onto the device by the MD user (local authorized administrator of the device), the platform will verify the signature from the Google Play Store/Apple Store/UEM Server store. Secure communication between the mobile device and the stores is handled by the underlying platform for each Hub Agent.

8.7 TOE Access

8.7.1 [MDMPP] FTA_TAB.1

[SERVER] The UEM Server supports the ability to display a configurable warning banner on the Admin Console and the Self-Service Portal login pages. The warning banner will be displayed to both Administrators and users prior to authenticating to the UEM Server on their respective interfaces. The

warning banner can be configured through the Admin Console Warning Banner tab by an Authorized Administrator.

8.8 Trusted Path/Channels

8.8.1 [MDMPP] FTP_ITC_EXT.1

[SERVER] The TOE has a communication channel between the UEM Server and one or more the Hub Agents for iOS and Android mobile devices. The UEM Server relies on its underlying platform to protect all data from disclosure and modification transferred over this internal communication channel. This communication channel is established once a mobile device has been fully enrolled into management, is logically distinct from other communication channels, and is specified under FPT_ITT.1(2).

[AGENT] The iOS and Android Hub Agents are internal to the TOE. The iOS and Android Hub Agents rely on their underlying platforms to protect all data from disclosure and modification transferred over this internal communication channel.

8.8.2 [MDMPP] FTP_ITC.1(1)

[SERVER] The UEM Server communicates with third-party systems that reside in the Operational Environment via trusted channels. In the evaluated configuration, the UEM Server connects with:

- the Syslog Audit Server using TLS v1.2 to encrypt the audit data that traverses the channel, and
- the AD/LDAP authentication server using TLS v1.2 for device enrollment using LDAP and to send authentication requests for an Administrator attempting to authenticate to the Admin Console.

The use of these protocols to establish trusted channels ensures that data in transit will be protected and not subjected to unauthorized modification or disclosure. During TLS session establishment, the UEM Server's platform will validate the third-party systems' presented X.509v3 certificates to validate their identities. If the third-party system is configured for mutual authentication, the UEM Server's identity is validated through its X.509v3 certificate presented during TLS session establishment. In the evaluated configuration, the UEM Server's platform will be configured to allow only the following ciphersuites:

- TLS_RSA_WITH_AES_128_CBC_SHA256 as defined in RFC 5246
- TLS_RSA_WITH_AES_256_CBC_SHA256 as defined in RFC 5246
- TLS_ECDHE_ECDSA_WITH_AES_128_CBC_SHA256 as defined in RFC 5289
- TLS_ECDHE_ECDSA_WITH_AES_256_CBC_SHA384 as defined in RFC 5289
- 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_256_GCM_SHA384 as defined in RFC 5289

The MAS Server is logically integrated with the UEM Server. As a result of this, all remote communications that the MAS Server requires to operate are identical to those described above.

8.8.3 [MDMPP] FTP_TRP.1(1)

[SERVER] The UEM Server's platform uses HTTPS/TLS (TLS v1.2) to provide a trusted communication between the UEM Server and Administrators attempting to connect to the Admin Console for the purposes

of remote administration. These protocols are used to protect the data traversing the channel from disclosure and/or modification. The UEM Server's identity is validated through its X.509v3 certificate presented during TLS session establishment and the Administrator's identity is validated through their authentication credentials presented to the UEM Server. In the evaluated configuration, the UEM Server's platform will be configured to allow only the following ciphersuites:

- TLS_RSA_WITH_AES_128_CBC_SHA256 as defined in RFC 5246
- TLS_RSA_WITH_AES_256_CBC_SHA256 as defined in RFC 5246
- TLS_ECDHE_ECDSA_WITH_AES_128_CBC_SHA256 as defined in RFC 5289
- TLS_ECDHE_ECDSA_WITH_AES_256_CBC_SHA384 as defined in RFC 5289
- 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_256_GCM_SHA384 as defined in RFC 5289

8.8.4 **[MDMPP] FTP_TRP.1(2)**

[SERVER] The UEM Server's platform uses HTTPS/TLS (TLS v1.2) to provide a trusted communication between the UEM Server and users. These protocols are used to protect the data traversing the channel from disclosure and/or modification. This communication path is used to connect to the Self-Service Portal for the purposes of remote device registration and other self-service tasks as well as the enrollment of the iOS and Android Hub Agents into the TOE. The UEM Server's identity is validated through its X.509v3 certificate presented during TLS session establishment and the user's identity is validated through their authentication credentials presented to the UEM Server. In the evaluated configuration, the UEM Server's platform will be configured to allow only the following ciphersuites:

- TLS_RSA_WITH_AES_128_CBC_SHA256 as defined in RFC 5246
- TLS_RSA_WITH_AES_256_CBC_SHA256 as defined in RFC 5246
- TLS_ECDHE_ECDSA_WITH_AES_128_CBC_SHA256 as defined in RFC 5289
- TLS_ECDHE_ECDSA_WITH_AES_256_CBC_SHA384 as defined in RFC 5289
- 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_256_GCM_SHA384 as defined in RFC 5289

[AGENT] The iOS and Android Hub Agents rely on their underlying platforms to provide the HTTPS/TLS communication path and to validate the UEM Server's X.509v3 certificate. After the enrollment of an iOS or Android Hub Agent into the TOE is complete, the initial connection handled by the communication path described above is closed and all future communication between the Hub Agent and UEM Server is governed by the internal channel described under the FPT_ITT.1(2) requirement.

9 Appendix A: List of Third-Party Libraries

9.1 Windows Server 2016 Libraries

- 7-zip.commandline
- adaptivasdk
- advapi32.dll
- airwatch.build
- animate.css
- antlr
- antlr3
- antlr3.runtime.pcl
- asp.net_mvc
- asyncex
- autofixture
- automapper
- aw.specflowplugin
- bogus
- bootstrap
- bouncycastle
- bundletransformer.autoprefixer
- bundletransformer.core
- bundletransformer.less
- cete.dynamicpdf
- clarity-angular
- clarity-icons
- clarity-ui
- clearscript.v8
- cleditor
- cleditor_wysiwyg_html_editor
- commandlineparser
- common.logging
- common.logging.core
- commonservicelocator
- componentspace.saml2
- confluent.kafka
- confluent.kafka.local
- consul
- costura.fody
- crypt32.dll
- d3
- dapper
- dataconnectiondialog
- DesEncrypt32.dll
- DesEncrypt64.dll
- dotcmis
- dotless
- dotless.core
- dotnetzip
- dotnetzip.reduced
- dynamicpdf
- edtftpnet-pro
- elasticsearch.net
- ember.js
- enterpriselibrary.common
- enterpriselibrary.data
- enterpriselibrary.data.sqlce
- enterpriselibrary.exceptionhandling
- enterpriselibrary.exceptionhandling.logging
- enterpriselibrary.exceptionhandling.wcf
- enterpriselibrary.logging
- enterpriselibrary.logging.database
- enterpriselibrary.policyinjection
- enterpriselibrary.transientfaulthandling
- enterpriselibrary.validation
- enterpriselibrary.validation.integration.aspnet
- enterpriselibrary.validation.integration.wcf
- enterpriselibrary.validation.integration.winforms
- entityframework
- entrustv9
- enyimmemcached
- epplus
- exceldatareader
- expressiveannotations
- fare
- fluentassertions
- fluentnest

- fluentvalidation
- fody
- fodycecil
- google.apis
- google.apis.admin.directory.directory_v1
- Google.apis.androidenterprise.v1
- google.apis.auth
- google.apis.core
- google.gdata.apps
- google.gdata.client
- google.gdata.extensions
- google.protobuf
- google.protobuf.tools
- gridstack
- handlebars.net
- hangfire
- hangfire.core
- hangfire.sqlserver
- hangfire.sqlserver.msmq
- hangfire.sqlserver.rabbitmq
- hiqpdf
- htmlagilitypack
- http2dotnet
- http2dotnet.hpack
- icomoon
- icsharpcode.sharpziplib.dll
- identityguardadminservicev8api
- identityguardcommonfailoverapi
- iesi.collections
- installengine
- interop.cert
- interop.certadminlib
- interop.certclientlib
- javascriptengineswitcher.core
- javascriptengineswitcher.msie
- javascriptengineswitcher.v8
- jetbrains.dotcover.commandlinetools
- jetbrains.dotmemoryunit
- jetbrains.microsoft.deployment.windows installer
- jquery
- jquery.cookie
- jquery.cycle
- jquery.dirtyforms
- jquery.facebox
- jquery.fancytree.combined
- jquery.form
- jquery.hoverintent
- jquery.scrollto
- jquery.tablesorter
- jquery.ui.combined
- jquery.validation
- jquery_colorbox
- jquery-bbq
- jquery-cookie-plugin
- jquery-cycle-plugin
- jquery-form-plugin
- jquery-ip
- jquery-multiselect
- jqueryserializeobject
- jquery-templating-plugin
- jquerytreetableplugin
- jquery-ui
- jquery-url-parser
- jquery-visualize
- jre
- js-cookie
- JSON Web Token Handler
- json4processing
- justmock
- kerberos
- Kernel32
- kernel32.dll
- knockout.contextmenu
- knockout.mapping
- knockoutjs
- librdkafka.redist
- log4net
- mailsystem.net-trunk
- managedesent
- marvin.jsonpatch
- microsoft.applicationinsights
- microsoft.applicationinsights.agent.intercept

- microsoft.applicationinsights.dependencycollector
- microsoft.applicationinsights.javascript
- microsoft.applicationinsights.perfcountercollector
- microsoft.applicationinsights.web
- microsoft.applicationinsights.windowsserver
- microsoft.applicationinsights.windowsserver.telemetrychannel
- microsoft.asp.net-universal-providers-core-libraries
- microsoft.aspnet.cors
- microsoft.aspnet.mvc
- microsoft.aspnet.providers
- microsoft.aspnet.providers.core
- microsoft.aspnet.razor
- microsoft.aspnet.web.optimization
- microsoft.aspnet.webapi
- microsoft.aspnet.webapi.client
- microsoft.aspnet.webapi.core
- microsoft.aspnet.webapi.cors
- microsoft.aspnet.webapi.helppage
- microsoft.aspnet.webapi.owin
- microsoft.aspnet.webapi.owinselfhost
- microsoft.aspnet.webapi.webhost
- microsoft.aspnet.webpages
- microsoft.azure.activedirectory.graphclient
- microsoft.bcl
- microsoft.bcl.async
- microsoft.bcl.build
- microsoft.codeanalysis.analyzers
- microsoft.codeanalysis.common
- microsoft.codeanalysis.csharp
- microsoft.codeanalysis.csharp.scripting
- microsoft.codeanalysis.scripting.common
- microsoft.codedom.providers.dotnetcompilerplatform
- microsoft.csharp
- microsoft.data.edm
- microsoft.data.odata
- microsoft.data.services.client
- microsoft.database.collections.generic
- microsoft.deployment.compression
- microsoft.exchange.webservices
- microsoft.extensions.logging.abstractions
- microsoft.identitymodel
- microsoft.identitymodel.clients.activedirectory
- microsoft.identitymodel.jsonwebtokens
- microsoft.identitymodel.logging
- microsoft.identitymodel.tokens
- microsoft.jquery.unobtrusive.ajax
- microsoft.jquery.unobtrusive.validation
- microsoft.net.compilers
- microsoft.net.http
- microsoft.netcore.platforms
- microsoft.netcore.targets
- microsoft.owin
- microsoft.owin.host.httplistener
- microsoft.owin.host.systemweb
- microsoft.owin.hosting
- microsoft.owin.testing
- microsoft.reportviewer.common
- microsoft.reportviewer.webforms
- microsoft.sharepoint.client
- microsoft.sharepoint.client.runtime
- microsoft.sqlserver.dacfx.x64
- microsoft.tpl.dataflow
- microsoft.web.administration
- microsoft.web.infrastructure
- microsoft.web.xdt
- microsoft.wim
- microsoft.win32.primitives
- Microsoft.WindowsAPICodePack
- microsoftmvcjqueryvalidation.js-from-microsoft-asp.net-mvc
- microsoft-sql-server
- microsoftwebmvc
- mimekit
- mimekitlite

- modernizr
- moq
- Mpr.dll
- mraspect.fody
- msbuild.sonarqube.runner.tool
- msiejavascriptengine
- msie-javascript-engine-for.net
- msipc.dll
- Mssign32.dll
- mstest.testadapter
- mstest.testframework
- mvc2futures
- nest
- netstandard.library
- newtonsoft.json
- ngx-clipboard
- nhibernate
- nito.asyncex
- nlipsum
- nlog
- nlog.targets.syslog
- node.js
- nsdepcop
- nuget.core
- owin
- png-image-encoder-in-c#
- polly
- qrcode.net
- quartz
- quartz-scheduler
- rabbitmq.client
- randomdatagenerator.net
- requirejs
- reset.css
- respond
- restease
- restsharp
- selenium.phantomjs.webdriver
- selenium.support
- selenium.webdriver
- selenium.webdriver.chromedriver
- serialization.plists
- servicestack.common.signed
- servicestack.interfaces
- servicestack.redis.signed
- servicestack.text.signed
- sharpziplib
- simmetrics.net
- sonaranalyzer.csharp
- specflow
- specflow.customplugin
- specflow.tools.msbuild.generation
- specflow.xunit
- starksoft.net.proxy
- stylecop
- stylecop.analyzers
- stylecop.msbuild
- stylecop.msbuild_modified
- svg
- swashbuckle
- swashbuckle.core
- Syroot.Windows.IO.KnownFolders
- syslognet.client_aw
- system.buffers
- system.collections
- system.collections.concurrent
- system.componentmodel.primitives
- system.componentmodel.typeconverter
- system.diagnostics.debug
- system.diagnostics.fileversioninfo
- system.diagnostics.stacktrace
- system.diagnostics.tools
- system.diagnostics.tracing
- system.dynamic.runtime
- system.globalization
- system.identitymodel.tokens.jwt
- system.io
- system.io.compression
- system.io.filesystem
- system.io.filesystem.primitives
- system.linq
- system.linq.dynamic.core
- system.linq.expressions
- system.management.automation

- system.net.http
- system.net.http.formatting.extension
- system.net.primitives
- system.objectmodel
- system.reactive.core
- system.reactive.interfaces
- system.reactive.linq
- system.reactive.platformservices
- system.reflection
- system.reflection.extensions
- system.resources.resourcemanager
- system.runtime
- system.runtime.compilerservices.unsafe
- system.runtime.extensions
- system.runtime.interopservices
- system.runtime.interopservices.runtimeinformation
- system.runtime.numerics
- system.runtime.serialization.plists
- system.runtime.serialization.primitives
- system.security.cryptography.encoding
- system.security.cryptography.x509certificates
- system.spatial
- system.text.encoding
- system.text.encoding.extensions
- system.text.regularexpressions
- system.threading
- system.threading.tasks
- system.threading.tasks.dataflow
- system.threading.tasks.extensions
- system.threading.thread
- system.threading.timer
- system.valuetuple
- system.web.providers
- system.web.providers.core
- system.web.routing
- system.xml.readerwriter
- system.xml.xdocument
- system.xml.xpath
- system.xml.xpath.xdocument
- twilio.api
- underscore.js
- unity
- unity.interception
- urlmon.dll
- Use Task.Runin synchronous method to avoid deadlock waiting on async
- wcf-service-with-wshttpbinding-manipulating-http-request-headers
- webactivatorex
- webgrease
- wintrust.dll
- wiremock.net
- wix
- WiX Toolset
- xpath2
- xpath2.extensions
- xunit
- xunit.abstractions
- xunit.analyzers
- xunit.assert
- xunit.core
- xunit.extensibility.core
- xunit.extensibility.execution
- xunit.runner.console
- xunit.runner.msbuild
- xunit.runner.visualstudio
- zlib.portable.signed

9.2 Android 9 Libraries

- acsdk.jar
- Anko
- apache-commons-modules-repackaged
- apache-commons-net
- AppAuth-Android
- chilkat-ftp-software
- com.android.support:multidex
- com.crashlytics.sdk.android:crashlytics
- com.google.dagger:dagger
- com.mixpanel.android:mixpanel-android

- com.squareup.picasso picasso
- cz.msebera.android:httpclient
- Global Protect
- Google-DPC-lib
- google-play-services
- gson
- guava
- httpclient
- io.reactivex.rxjava2:rxandroid
- io.reactivex.rxjava2:rxjava
- jcifs
- jzlib
- kotlin-stdlib-jre7
- krb5
- littleproxy
- mixpanel
- netty
- OpenSSL
- org.apache.commons:io
- org.greenrobot:eventbus
- org.greenrobot:greendao
- org.simpleframework:simple-xml
- proxy-vole
- Rhino 1.7
- rsa-AA-Mobile-SDK-Android
- slf4j
- sqlcipher-for-android
- stringencrypter.java
- support-libraries-for-android-api
- zxing

9.3 iOS 12 Libraries

- Alamofire
- AlamofireImage
- atomictools
- CocoaLumberjack
- code-from-ios-developerscookbook-uidevice-hardware.h
- commoncrypto
- deusty-llc
- FastSocket
- HexColors
- how-do-i-base64-encode-(decode)-in-c
- HttpStatusCodes
- ios-5-programming-pushing-the-limits
- iphone-developer's-cookbook
- JSQCoreDataKit
- KeychainAccess
- kissxml
- lottie
- MixPanel-swift
- OpenSSL
- plcrashreporter
- reachability
- rncryptorold.m
- RSA Adaptive Authentication Module
- sfhfkeychainutils
- Starscream
- STWebarchive
- uiimage-resize-then-crop