Endpoint Manager Mobile (EPMM) System 12 Security Target

Version 3.5

13 October 2025

10377 South Jordan Gateway
Suite 110
South Jordon, Utah 84095

2400 Research Blvd
Suite 395
Rockville, MD 20850

Revision History

Version	Date	Changes
Version 0.1	April 22, 2024	Draft MDM
	June 11, 2024	GAP responses
Version 0.2	June 24, 2024	Vendor Response Update Related to GAP
Version 0.3	July 11, 2024	Vendor response Updates Related to GAP/ST comments
Version 0.4	September 10, 2024	Comment version bump.
Version 0.5	September 10, 2024	Check-in related updates. Testing related consistency issues.
Version 0.6	September 26, 2024	Updates related to comment round. New comments related to early testing results. Last round before check-in.
Version 0.7	September 30, 2024	Updates related to comment round with vendor. Clean up and self-review before check-in.
Version 0.8	October 4, 2024	Peer reviewer updates before check-in.
Version 2.2	January 14, 2025	Addressed issues.
Version 2.9	August 12, 2025	Addressed issues.
Version 2.10	August 20, 2025	Addressed reviewers comments
Version 3.0	August 26, 2025	Addressed reviewers comments
Version 3.1	September 9, 2025	Addressed reviewers comments
Version 3.2	September 10, 2025	Addressed reviewers comments
Version 3.3	October 3, 2025	Addressed ECRs.
Version 3.4	October 09, 2025	Addressed ECRs.
Version 3.5	October 13,2025	Addressed comments.

Table of Contents

1 INTRODUCTION 1

1.1 Security Target and TOE Reference 1
1.2 Product Overview 1
1.3 TOE Overview 2
1.4 TOE Type 2
1.5 Use Case 2
1.6 TOE Description 2
1.6.1 Physical Boundary 3
1.6.2 Logical Boundary 11
1.7 TOE Documentation 12
1.8 Product Functionality not Included in the Scope of the Evaluation 12

2 CONFORMANCE CLAIMS 14

2.1 CC Conformance Claims 14
2.2 Protection Profile Conformance 14
2.3 Conformance Rationale 14
2.3.1 Technical Decisions 14

3 SECURITY PROBLEM DEFINITION 18

3.1 Threats 18
3.2 Assumptions 18
3.3 Organizational Security Policies 20

4 SECURITY OBJECTIVES 21

4.1 Security Objectives for the TOE 21
4.2 Security Objectives for the Operational Environment 21

5 SECURITY REQUIREMENTS 23

5.1 Conventions 23
5.2 Extended Functional Requirements 23
5.3 Security Functional Requirements List 24
5.4 Security Functional Requirements (SFRs) 26
5.4.1 Security Audit (FAU) 26
5.4.2 Communication (FCO) 33
5.4.3 Cryptographic Support (FCS) 34
5.4.4 Identification and Authentication (FIA) 39
5.4.5 Security Management (FMT) 41
5.4.6 Protection of the TSF (FPT) 44
5.4.7 Trusted Path/Channels (FTP) 45
5.5 TOE SFR Dependencies Rationale for SFRs 47
5.6 Security Assurance Requirements (SARs) 47
5.7 Assurance Measures 47

6 TOE SUMMARY SPECIFICATION 49

6.1 Security Audit (FAU) 49
6.1.1 FAU_ALT_EXT.1 Server Alerts (MDM_PP) 49
6.1.2 FAU_ALT_EXT.2 Agent Alerts (AGT_PMM) 49
6.1.3 FAU_CRP_EXT.1 Support for Compliance Reporting of Mobile Device Configuration (MDM_PP) 50
6.1.4 FAU_GEN.1(1) Audit Data Generation (MDM_PP) 50

6.1.5 FAU_GEN.1(2)/AGT Audit Data Generation (AGT_PPM)....51
6.1.6 FAU_GEN.1(2)/SRV Audit Generation (MAS Server) (MDM_PP)....52
6.1.7 FAU_NET_EXT.1 Network Reachability Review (MDM_PP)....52
6.1.8 FAU_SAR.1 Audit Review (MDM_PP)....52
6.1.9 FAU_SEL.1(2) Security Audit Event Selection (AGT_PPM)....53
6.1.10 FAU_STG_EXT.1 External Trail Storage (MDM_PP) and FAU_STG_EXT.2 Audit Event Storage (MDM_PP)....54
6.2 COMMUNICATION (FCO)....55
6.2.1 FCO_CPC_EXT.1 Component Registration Channel Definition (MDM_PP)....55
6.3 CRYPTOGRAPHIC SUPPORT (FCS)....55
6.3.1 FCS_CKM.1 Cryptographic Key Generation (MDM_PP)....55
6.3.2 FCS_CKM.2 Cryptographic Key Establishment (MDM_PP)....55
6.3.3 FCS_CKM_EXT.4 Cryptographic Key Destruction (MDM_PP)....56
6.3.4 FCS_COP.1.1(1) Cryptographic Operation (Confidentiality Algorithms) (MDM_PP)....57
6.3.5 FCS_COP.1(2) Cryptographic Operation (Hashing Algorithms) (MDM_PP)....58
***6.3.6** FCS_COP.1(3) Cryptographic Operation (Signature Algorithms) (MDM_PP)....58*
6.3.7 FCS_COP.1(4) Cryptographic Operation (Keyed-Hash Message Authentication) (MDM_PP)....59
6.3.8 FCS_HTTPS_EXT.1 HTTPS Protocol`(MDM_PP)....59
6.3.9 FCS_IV_EXT.1 Initialization Vector Generation (MDM_PP)....59
6.3.10 FCS_RBG_EXT.1/AGT Extended: Random Bit Generation (MDM_PP)....60
6.3.11 FCS_RBG_EXT.1/SRV Extended: Random Bit Generation (MDM_PP)....60
6.3.12 FCS_STG_EXT.1 Cryptographic Key Storage (MDM_PP) and FCS_STG_EXT.2 Encrypted Cryptographic Key Storage (MDM_PP)....60
6.3.13 FCS_STG_EXT.1(2) Cryptographic Key Storage (AGT_PPM)....60
6.3.14 FCS_TLS_EXT.1 TLS Protocol (TLS_PKG)....60
6.3.15 FCS_TLSC_EXT.1 TLS Client Protocol (TLS_PKG)....62
6.3.16 FCS_TLSC_EXT.2 TLS Client Support for Mutual Authentication (TLS_PKG)....63
6.3.17 FCS_TLSC_EXT.5 TLS Client Support for Supported Groups Extensions (TLS_PKG)....64
6.3.18 FCS_TLSS_EXT.1 TLS Server Protocol (TLS_PKG)....64
6.3.19 FCS_TLSS_EXT.2 TLS Server Support for Mutual Authentication (TLS_PKG)....64
6.4 IDENTIFICATION AND AUTHENTICATION (FIA)....65
6.4.1 FIA_CLI_EXT.1 X.509 Unique Certificate (MDM_PP)....65
6.4.2 FIA_ENR_EXT.1 Enrollment of Mobile Device into Management (MDM_PP)....65
6.4.3 FIA_ENR_EXT.2 Agent Enrollment of Mobile Device into Management (AGT_PPM)....67
6.4.4 FIA_UAU.1 Timing of Authentication (MDM_PP)....67
6.4.5 FIA_X509_EXT.1(1) X.509 Certificate Validation (MDM_PP)....68
6.4.6 FIA_X509_EXT.2 X.509 Certificate Authentication (MDM_PP)....69
6.5 SECURITY MANAGEMENT (FMT)....70
6.5.1 FMT_MOF.1(1) Management of Functions Behaviour (MDM_PP)....70
6.5.2 FMT_MOF.1(2) Management of functions Behaviour (Enrollment) (MDM_PP)....72
6.5.3 FMT_MOF.1(3) Management of Functions in (MAS Server Downloads) (MDM_PP)....72
6.5.4 FMT_POL_EXT.1 Trusted Policy Update (MDM_PP) and FMT_POL_EXT.2 Agent Trusted Policy Update (AGT_PPM)....73
6.5.5 FMT_SMF.1(1) Specification of Management Functions (Server configuration of Agent) (MDM_PP) 73
6.5.6 FMT_SMF.1(2) Specification of Management Functions (Server configuration of Server) (MDM_PP) 81
6.5.7 FMT_SMF.1(3) Specification of Management Functions (MAS Server) (MDM_PP)....81
6.5.8 FMT_SMF_EXT.4 Specification of Management Functions (AGT_PPM)....81
6.5.9 FMT_SMR.1(1) Security Management Roles (MDM_PP)....82
6.5.10 FMT_SMR.1(2) Security Management Roles (MAS Server) (MDM_PP)....86
6.5.11 FMT_UNR_EXT.1 User Unenrollment Prevention (AGT_PPM)....86

6.6 PROTECTION OF THE TSF (FPT) 87
6.6.1 FPT_API_EXT.1 Use of Supported Services and API's (MDM_PP) *87*
6.6.2 FPT_ITT.1(2) Internal TOE TSF Data Transfer (MDM Agent) (MDM_PP) *88*
6.6.3 FPT_LIB_EXT.1/AGT Use of Third Party Libraries (MDM_PP) *88*
6.6.4 FPT_LIB_EXT.1/SRV Use of Third Party Libraries (MDM_PP) *89*
6.6.5 FPT_TST_EXT.1 Functionality Testing (MDM_PP) *89*
6.6.6 FPT_TUD_EXT.1 Trusted Update (MDM_PP) *90*
6.7 TRUSTED PATH/CHANNELS (FTP) 91
6.7.1 FTP_ITC.1(1) Inter-TSF Trusted Channel (Authorized IT Entities) (MDM_PP) *91*
6.7.2 FTP_ITC_EXT.1 Trusted Channel (MDM_PP) *92*
6.7.3 FTP_TRP.1(1) Trusted Path (for Remote Administration) (MDM_PP) *92*
6.7.4 FTP_TRP.1(2) Trusted Path (for Enrollment) (MDM_PP) *92*
6.8 EPMM MDM CRYPTOGRAPHY 93
6.8.1 EPMM MDM Cryptographic Libraries *93*
6.8.2 CAVP Algorithm Certificate Details *93*
6.9 DISTRIBUTED TOE SFR ALLOCATION 99
7 TERMS, ACRONYMS, AND ABBREVIATIONS 103
APPENDIX A 107
APPENDIX B 115

List of Figures

Figure 1: The TOE and the MDM Server's Operational Environment 4
Figure 2: The TOE and the MDM Agent's Operational Environment 5
Figure 3: Enrollment Process 66

List of Tables

Table 1: TOE/ST Identification 1
Table 2: The Ivanti EPMM Device/Host/Server Operational Environment 5
Table 3: Communication Path Lines SFR Claims 7
Table 4: MDM Server Hardware Detail (Operational Environment) 9
Table 5: MDM Server Software Detail (Operational Environment) 9
Table 6: The MDM Server TOE Component Build 9
Table 7: MDM Agent Host Hardware Detail (Operational Environment) 10
Table 8: The MDM Agent TOE Component Build 11
Table 9: Relevant Technical Decisions (TDs) 15
Table 10: Threats 18
Table 11: Assumptions 19
Table 12: Organizational Security Policies (OSPs) (*Applied TD0497*) 20
Table 13: Security Objectives for the TOE (*Addressed TD0497*) 21
Table 14: Security Objectives for the Operational Environment 22
Table 15: Security Functional Requirements (SFRs) 24
Table 16: MDM Server Security Functional Requirements and Auditable Events 28
Table 17: EPMM Agent *(AGT_PPM)* Security Functional Requirements and Auditable Events (addressed TD0660) 30
Table 18: Reference and IV Requirements for NIST-approved Cipher Modes 36
Table 19: Security Assurance Requirements 47
Table 20: TOE Security Assurance Measures 48
Table 21: Keys and CSPs 56

Table 22: HMAC Details 59
Table 23: EPMM TLS Protocol Implementation 61
Table 24: EPMM MDM Server Management Functions and Roles 70
Table 25: MDF to MDM FMT_SMF.1 Mapping 78
Table 26: EPMM Permission to Role Mappings 83
Table 27: Ivanti EPMM Cryptographic Library CAVP Details 93
Table 28: CAVP Algorithm Certificate References 93
Table 29: Distributed TOE SFR Allocation 99
Table 30: Terms 103
Table 31: Acronyms and Abbreviations 103
Table 32: Audit Event Type: Administrative Action: MDM Server 107
Table 33: MDM Server Security Functional Requirements and Auditable Events 110
Table 34: Events and messages 111
Table 35: EPMM Agent *(AGT_PPM)* Security Functional Requirements and Auditable Events (addressed TD0660) 113

1 Introduction

This Security Target (ST) serves as the basis for the Common Criteria (CC) evaluation and identifies the Target of Evaluation (TOE), the scope of the evaluation, and the assumptions made throughout. This document will also describe the intended operational environment of the TOE, and the functional and assurance requirements that the TOE meets.

1.1 Security Target and TOE Reference

This section provides the information needed to identify and control the TOE and the ST.

Table 1: TOE/ST Identification

Category	Identifier
ST Title	Ivanti Endpoint Manager Mobile (EPMM) System 12 Security Target
ST Version	V3.4
ST Date	13 October 2025
ST Author	Acumen Security
TOE Identifier	Endpoint Manager Mobile (EPMM) System 12
TOE Version	84 [build] 12.3.1.0 (Server virtual appliance software) and Version 12 (Android Agents)
TOE Developer	Ivanti, Inc.
Key Words	Mobile Device Management

1.2 Product Overview

The Ivanti Endpoint Manager Mobile (EPMM) System is a distributed TOE and includes two software builds: the Ivanti MDM Server and the Ivanti MDM Agent. An EPMM implementation supports one instance of the MDM Server and one or more instances of the MDM Agent.

Ivanti Endpoint Manager Mobile (EPMM) is mobile management software that enables Administrators to set policies for mobile devices, applications, and content. EPMM enables mobile device management, mobile application management, and mobile content management capabilities and includes the following tools:

- Mobile Device Management (MDM) – Provides tools that manage and track mobile devices.
- Mobile Application Management (MAM) – Allows access to specific applications.
- Mobile Content Management (MCM) - Supports creating, organizing, governing, and storing digital assets.

EPMM enables administrators to securely manage the lifecycle of mobile devices and mobile applications, from registering a device with Ivanti EPMM, to retiring the device from EPMM management. Users of an Ivanti EPMM managed device can securely access corporate data, email, and mobile apps that are controlled and distributed using Ivanti EPMM.

1.3 TOE Overview

This Security Target is for the Common Criteria evaluation of the Ivanti Endpoint Manager Mobile (EPMM) System version 12 against the *Protection Profile for Mobile Device Management*, Version 4.0, *PP-Module for MDM Agents*, Version 1.0, and *Functional Package for Transport Layer Security (TLS)*, Version 1.1.

The Target of Evaluation (TOE) is the Ivanti Endpoint Mobile Manager (EPMM) 12 System (Version 12). The TOE (EPMM) is a distributed TOE and consists of two software components: the EPMM MDM Server and the EPMM MDM Agent (also referred to as MDM Server and MDM Agent).

The MDM Agent is installed on Android devices as an application that is supplied by Ivanti.

This Ivanti CC evaluation includes the Mobile Device Management (MDM) and the Mobile Application Management (MAM) functionality of the Ivanti EPMM. The Mobile Content Management (MCM) functionality of the EPMM is excluded from the CC evaluation. Additionally, the Target of Evaluation (TOE) includes Mobile Application Store (MAS) functionality that hosts applications, authenticates Agents, and securely transmits applications to enrolled mobile devices.

The MDM Server is a software application deployed on a virtual host. The MDM Server manages and monitors one or more instances of an MDM Agent. The MDM Agent is an application running on an Android device. This evaluation includes MDM Agent running on two different Android devices and operating system version:

- Galaxy S22 Ultra 5G running Android 13
- Galaxy S23 Ultra 5G running Android 14

Details of the evaluated devices are in Section 1.6.1 Physical Boundary below.

1.4 TOE Type

The TOE is a distributed mobile device management system and includes both the server software and the client software.

1.5 Use Case

The TOE's use case is [USE CASE 1] "Enterprise-owned device for general purpose enterprise use" as defined in MOD_MDM_AGENT_V1.0. [USE CASE 1] An Enterprise-owned device for general-purpose business use is commonly called Corporately Owned, Personally Enabled (COPE). This use case entails a significant degree of Enterprise control over configuration and software inventory. Enterprise administrators use an MDM product to establish policies on the mobile devices prior to user issuance. Users may use Internet connectivity to browse the web or access corporate mail or run Enterprise applications, but this connectivity may be under significant control of the Enterprise. The user may also be expected to store data and use applications for personal, non-enterprise use. The Enterprise administrator uses the MDM product to deploy security policies and query mobile device status.

1.6 TOE Description

The TOE is an MDM solution where the claimed security functions are implemented in a central MDM Server and distributed MDM Agents. The EPMM MDM Server integrates with backend enterprise IT systems and enables IT to define security and management policies for mobile apps.

The MDM Server is a software application that runs on an Intel x64 architecture server platform as a virtual system with Oracle Linux as the operating system (OS).

The MDM Server provides two administrative interfaces, each used for specific administrative actions. The Admin Portal provides capabilities for administrators to manage users, devices, policies and mobile device configurations. The System Manager provides administrative functions to control and configure the operation of the MDM Server including basic system configuration information (e.g., network addresses, date/time, hostname, e-mail settings), TOE updates, certificate management, TLS configuration, and export of various operational logs.

The MDM Agent, an application running on Galaxy S22 and S23 running Android 13 and 14 respectively, is an Android application. The MDM Agents work with the MDM Server to configure corporate email, Wi-Fi, VPN, and X.509 security certificates and to create a clear separation between personal and business information. Once installed, the MDM Agent creates a secure EPMM container that protects enterprise data and applications. This allows an MDM Server Administrator to selectively wipe only the enterprise data on the device if the user leaves or if the device falls out of compliance or is lost.

1.6.1 Physical Boundary

1.6.1.1 Physical Boundary and the Operational Environment

The TOE is a distributed TOE that includes two software builds, EPMM MDM Server and EPMM MDM Agent (MDMPP Client). The builds include software libraries that include the cryptographic libraries; the HTTP Server library that provides the Graphical User Interface (GUI) as well as other off-the-shelf (OTS) libraries. Each TOE component runs on an operating system provided by the Operational Environment (OE) and accesses the OE libraries by the well known APIs.

The MDM Server software build includes two cryptographic libraries: Ivanti MDM OpenSSL Component MDM OpenSSL and Ivanti MDM Bouncy Castle. Both libraries are CAVP certified and described in detail Section 6.8. As depicted in the figures below, the Ivanti MDM OpenSSL Component cryptographic library is used to support the two administrator interfaces, Admin Portal and System Manager; communicate with the Audit Server; and provides the communication channels to the MDM Agents. The MDM Bouncy Castle cryptographic library is used as an initial connection to the Audit Server and support data at rest, encrypting X.509 certificates stored in flat files.

The MDM Agent software build includes one cryptographic library Ivanti MDM Android Client OpenSSL Component that is CAVP certified and described in detail Section 6.8. Ivanti MDM Android Client OpenSSL Component provides the cryptographic function used for the communication channels to the MDM Server. The TOE relies on the Android device's Operational Environment's cryptographic library, Samsung BoringSSL Android 1.7 and Samsung BoringSSL Android 1.8 for Android 13 and Android 14 respectively, to encrypt data stored in Android's Truststore, provided by the OE.

The following two figures depict the two TOE components in their operational environment. They provide a rough diagram of the software TOE and identify the security enforcing software modules. The TOE is depicted in turquoise (hash and solid). The figures include a capital letter identifying each of the communication paths. Details of the communication paths is in Table 3.

Figure 1: The TOE and the MDM Server's Operational Environment

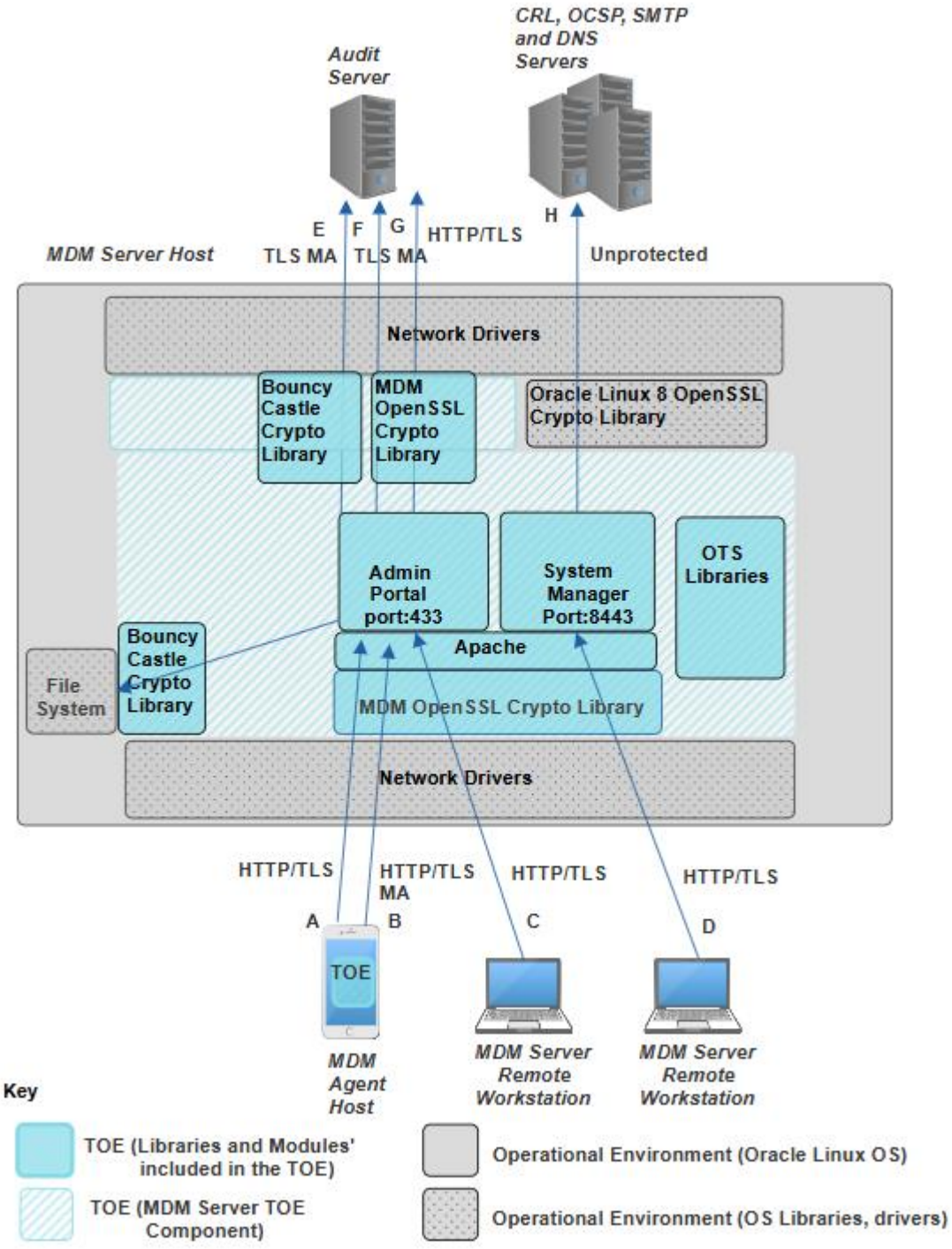
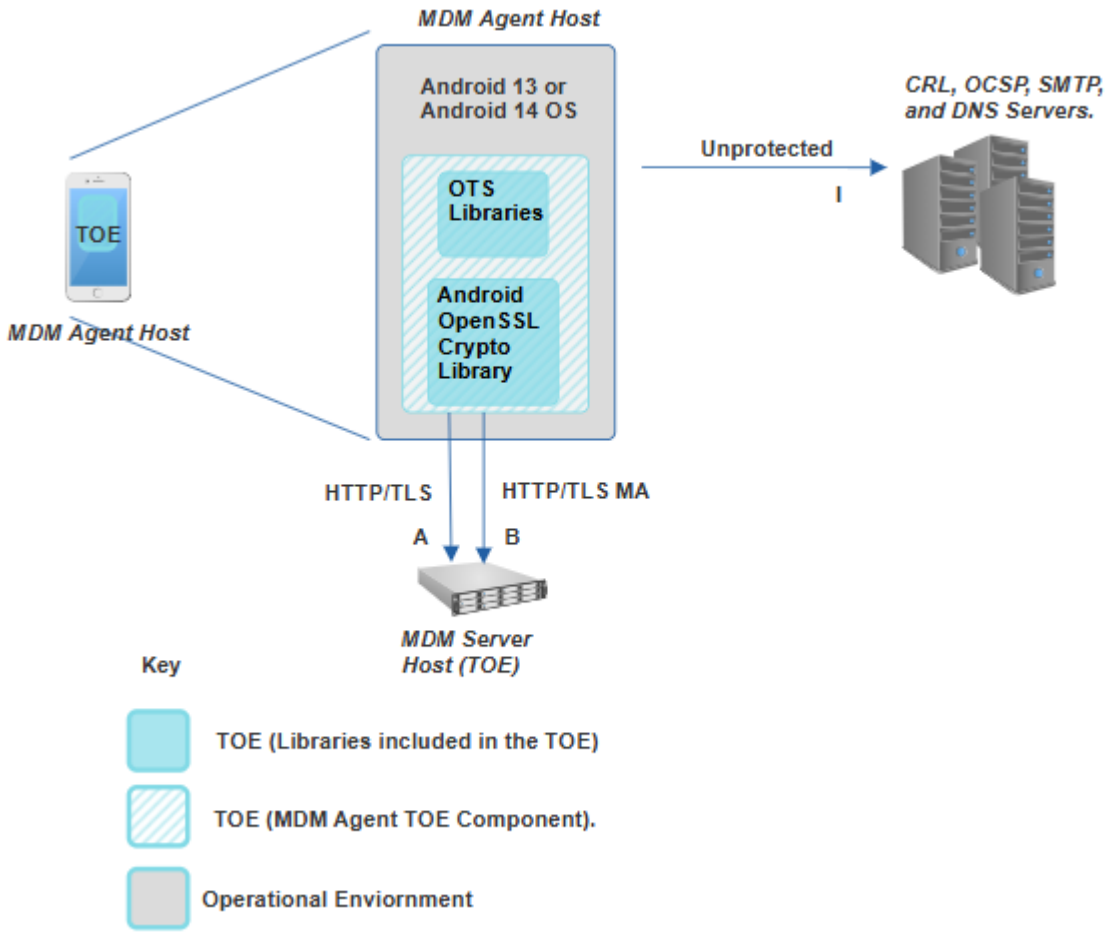


Figure 2: The TOE and the MDM Agent's Operational Environment



The following table identifies the servers, workstations, and hardware required for the EPMM MDM Server and MDM Agent components of the TOE.

Details of the figure's labeled communication paths is in Table 3.

Table 2: The Ivanti EPMM Device/Host/Server Operational Environment

Components	Mandatory /Optional	TOE Components	Description
CRL Servers	Mandatory	Both	A Certificate Revocation List (CRL) Server used to validate X.509 certificates. Communication is via an unprotected channel.
DNS Server(s)	Mandatory	Both	A Domain Name System (DNS) Server used to resolve Fully Qualified Domain Names (FQDNs) to IP addresses for certificate validation. A DNS Server is required in both TOE environments. The TOE components

Components	Mandatory /Optional	TOE Components	Description
			communicate with DNS Servers on an unprotected path.
MDM Agent Host	Mandatory	MDM Agent	The device hosting the MDM Agent component of the TOE. The host is a Galaxy S22 Ultra 5G device running Android 13 OS or a Galaxy S23 Ultra 5G device running Android 14 OS. The MDM Agent software component of the TOE must be installed on the host and configured in the CC evaluated configuration. Refer to Section 1.6.1.3 for specific details of the MDM Agent Host.
MDM Server Host	Mandatory	MDM Server	The device hosting the MDM Server software component of the TOE. The host is a Dell PowerEdge R640 Server with hypervisor VMware ESXi 7.02 running Oracle Linux 8.9 Operating System. The MDM Server software component of the TOE must be installed on the host and configured in the CC evaluated configuration. Refer to Section 0 for specific details of the MDM Server Host.
MDM Server Remote Workstation(s)	Mandatory	MDM Server	The MDM Server component of the TOE supports two GUI interfaces that enable Administrators to manage and monitor the TOE. Administrators access these interfaces via remote workstations using HTTP over TLS (HTTPS). The OE requires one or more remote workstations connecting to the MDM Server's Admin Portal administrator interface and the MDM Server's System Manager administrator interface.
OCSP Servers	Mandatory	Both	An Online Certificate Status Protocol (OCSP) Server used to validate X.509 certificates. Communication is via an unprotected channel.
SMTP Server	Mandatory	Both	The MDM Server supports sending Alerts if a specific event occurs. One type of Alerts is an email message. Therefore, both TOE components must have access to a Simple Mail Transfer Protocol Server (SMTP). Communication is via an unprotected channel.
Audit Server	Mandatory	MDM Server	A Audit Server is required by the MDM Server to enable the MDM Server to transfer

Components	Mandatory /Optional	TOE Components	Description
			audit Events. Communication is via mutual authenticated TLS. The Audit Server must support Syslog and HTTPS.

Table 3: Communication Path Lines SFR Claims

Line ID - PP_MDM_V4.0 or MOM_MDM_AGENT_V1.0 SFR Claim					
TLS Client ID	TLS SFR Claim	Client Crypto Library	TLS Server ID	TLS SFR Claim	Server Crypto Library
Line A - FTP_TRP.1(2) Trusted Path (for Enrollment) *(MDM_PP)* The initial HTTP over TLS connection initiated from the MDM Agent to the MDM Server to start enrollment.					
MDM Agent	FCS_TLSC_EXT.1 TLS Client Protocol *(TLS_PKG)*	Ivanti MDM Android Client OpenSSL Component 2.2.1	MDM Server	FCS_TLSS_EXT.1 TLS Server Protocol *(TLS_PKG)*	Ivanti MDM OpenSSL Component 1.1.1g
Line B - FPT_ITT.1(2) Internal TOE TSF Data Transfer (MDM Agent) *(MDM_PP)* The second HTTPS over TLS (MA) connection initiated by the MDM Agent to the MDM Server to complete enrollment. This is the protected channel used to transfer data between the two MDM components.					
MDM Agent	FCS_TLSC_EXT.1 TLS Client Protocol *(TLS_PKG)* FCS_TLSC_EXT.2 TLS Client Support for Mutual Authentication *(TLS_PKG)*	Ivanti MDM Android Client OpenSSL Component 2.2.1	MDM Server	FCS_TLSS_EXT.1 TLS Server Protocol *(TLS_PKG)* FCS_TLSS_EXT.2 TLS Server Support for Mutual Authentication *(TLS_PKG)*	Ivanti MDM OpenSSL Component 1.1.1g
Line C - FTP_TRP.1(1) Trusted Path (for Remote Administration) *(MDM_PP)* An HTTP over TLS connection from a remote workstation to the Admin Portal for Administrator GUI access.					
MDM Server Remote Workstation	N/A (OE)	N/A	MDM Server	FCS_TLSS_EXT.1 TLS Server Protocol *(TLS_PKG)*	Ivanti MDM OpenSSL Component 1.1.1g
Line D - FTP_TRP.1(1) Trusted Path (for Remote Administration) *(MDM_PP)* An HTTP over TLS connection from a remote workstation to the MDM Servers System Manager portal for Administrator GUI access.					

Line ID - PP_MDM_V4.0 or MOM_MDM_AGENT_V1.0 SFR Claim					
TLS Client ID	TLS SFR Claim	Client Crypto Library	TLS Server ID	TLS SFR Claim	Server Crypto Library
MDM Server Remote Workstation	N/A (OE)	N/A	MDM Server	FCS_TLSS_EXT.1 TLS Server Protocol *(TLS_PKG)*	Ivanti MDM OpenSSL Component 1.1.1g
Line E – FTP_ITC.1(1) Inter-TSF Trusted Channel (Authorized IT Entities) *(MDM_PP)* A TLS connection from the MDM Server to the Audit Server supporting Syslog. The MDM Server initially connects to the Syslog Server using Bouncy Castle in order to validate the server's certification. If validated, the MDM Server initiates Line F.					
MDM Server	FCS_TLSC_EXT.1 TLS Client Protocol *(TLS_PKG)* FCS_TLSC_EXT.2 TLS Client Support for Mutual Authentication *(TLS_PKG)*	Ivanti MDM Bouncy Castle Component 1.0.2.4	Audit Server (Syslog)	N/A (OE)	N/A
Line F - FTP_ITC.1(1) Inter-TSF Trusted Channel (Authorized IT Entities) *(MDM_PP)* A TLS connection from the MDM Server to the Audit Server supporting Syslog. The MDM Server automatically transfers MDM Server audit logs to the Syslog Server.					
MDM Server	FCS_TLSC_EXT.1 TLS Client Protocol *(TLS_PKG)* FCS_TLSC_EXT.2 TLS Client Support for Mutual Authentication *(TLS_PKG)*	Ivanti MDM OpenSSL Component 1.1.1g	Audit Server (Syslog)	N/A (OE)	N/A
Line G – FTP_ITC.1(1) Inter-TSF Trusted Channel (Authorized IT Entities) *(MDM_PP)* An HTTPS (HTTP over TLS) connection from the MDM Server to the Audit Server that enables the MDM Server to transfer fetched MDM Agent log files to the Audit Server.					
MDM Server	FCS_TLSC_EXT.1 TLS Client Protocol *(TLS_PKG)* FCS_TLSC_EXT.2 TLS Client Support for Mutual Authentication *(TLS_PKG)*	Ivanti MDM OpenSSL Component 1.1.1g	Audit Server	N/A (OE)	N/A
Line H – N/A Unprotected channel An unprotected channel for the MDM Server to communicate with remote OE Servers.					

Line ID - PP_MDM_V4.0 or MOM_MDM_AGENT_V1.0 SFR Claim					
TLS Client ID	TLS SFR Claim	Client Crypto Library	TLS Server ID	TLS SFR Claim	Server Crypto Library
MDM Server	N/A (unprotected)	N/A	CRL, OCSP, SMTP, and DNS Server	N/A (OE)	N/A
Line I – N/A Unprotected channel Unprotected channel for MDM Agent to communicate with remote OE Servers.					
MDM Agent	N/A (unprotected)	N/A	CRL, OCSP, SMTP, and DNS Server	N/A (OE)	N/A

1.6.1.2 The MDM Server Details

The MDM Server runs on a virtual system and is described in the following tables.

Table 4: MDM Server Hardware Detail (Operational Environment)

Model	Processor	GHz	Hypervisor
Dell PowerEdge R640 Server	Intel(R) Xeon(R) Gold 5215 CPU (Cascade Lake)	2.10Ghz	VMware ESXi 7.02

Table 5: MDM Server Software Detail (Operational Environment)

Item	Software
Operating System	Oracle Linux 8.9
Runtime Environment	Java SE Run Java SE Runtime Environment v8 (1.8) on Oracle Linux 8.9
Hypervisor	VMWare ESXi 7.02
Kernel	Red Hat Compatible Kernel (RHCK) kernel package kernel-4.18.0-80.el8

The MDM Server, Ivanti Endpoint Manager Mobile (Core), can be downloaded by customers from https://forums.ivanti.com/s/contactsupport?language=en_US and installed on compliant hardware listed above. Licenses are provided by Ivanti Secure via email. When a customer request is received, Ivanti will provide an authcode via email. Customers must register in https://forums.ivanti.com/s/contactsupport?language=en_US portal and generate the license string by providing Hardware id with earlier provided authcode. These auth codes are not reusable.

Table 6: The MDM Server TOE Component Build

Build Name	Cryptographic Libraries included in the Build	CAVP #	Other Libraries included in the Build
EPMM 12.3.1.0 Build 84	Ivanti MDM Bouncy Castle Component 1.0.2.4	#A6073	Apache 2.4

Build Name	Cryptographic Libraries included in the Build	CAVP #	Other Libraries included in the Build
	Ivanti MDM OpenSSL Component 1.1.1g	#A6074	OTS Libraries identified in Appendix B

Refer to Section 6.8 for complete information about the CAVP certified cryptographic libraries included in the TOE.

1.6.1.3 The MDM Agent Details

The MDM Agent consists of a software application deployed on one of two Android mobile devices.

Galaxy S22 Ultra 5G Android 13
Samsung Electronics Co., Ltd.

Galaxy S23 Ultra 5G Android 14
Samsung Electronics Co., Ltd.

The following tables include the Android device details.

Table 7: MDM Agent Host Hardware Detail (Operational Environment)

Manufacturer	Device Name	Models	Chipset Vendor	SoC	Microarchitecture
Samsung	Galaxy S22 Ultra 5G	SM-S908B, SM-S908B/DS, SM-S908U, SM-S908U1, SM-S908W, SM-S908N, SM-S9080, SM-S908E, SM-S908E/DS	Samsung	Exynos 2200 (AMD RDNA™ 3)	ARMv8
Samsung	Galaxy S23 Ultra 5G	SM-S918B, SM-S918B/DS, SM-S918U, SM-S918U1, SM-S918W, SM-S918N, SM-S9180, SM-S918E, SM-S918E/DS	Qualcomm	Snapdragon 8 Gen 2 Mobile Platform	ARMv8

The MDM Agent is obtained from Ivanti personnel and is referred to as the Federal Build (vs. the commercial build).

Table 8: The MDM Agent TOE Component Build

Build Name	Cryptographic Library included in the Build	CAVP #	Other Libraries included in the Build
MDMPP-MIClient-12.5.1.9.R	Ivanti MDM Android Client OpenSSL Component 2.2.1	#6402	Apache
			OTS Libraries identified in *Appendix B*

Refer to Section 6.8 for complete information about the CAVP certified cryptographic libraries included in the TOE.

1.6.2 Logical Boundary

The TOE provides the security functions required by the:

- *Protection Profile for Mobile Device Management*, Version 4.0, hereafter referred to as PP_MDM_V4.0.
- *PP-Module for MDM Agent*, Version 1.0, hereafter referred to as MOD_MDM_AGENT_V1.0.
- *Functional Package for Transport Layer Security (TLS)*, Version 1.1, hereafter referred to as PKG_TLS_v1.1.

1.6.2.1 Security Audit

The MDM Server has the capability to create and retain audit logs for significant security incidents in real-time. These logs are securely stored by the MDM Server and can be accessed for review by authorized administrators. Additionally, the MDM Server exports these audit logs to an Audit Server.

The MDM Agent possesses the capability to produce audit logs for events pertaining to security and can signal when it has been registered and when policies are effectively implemented onto it.

1.6.2.2 Cryptographic Support

Both the MDM Server and MDM Agent incorporate cryptographic modules equipped with certified algorithms to perform cryptographic operations. These operations encompass asymmetric key generation and exchange, encryption/decryption, cryptographic hashing, and keyed-hash message authentication. To support these functions adequate mechanisms for random bit generation, the destruction of keys and protected data are provided.

The cryptographic functionality is used to implement security communication protocols: TLS and HTTPS used for communication between the MDM Server and the MDM Agent, the Audit Server, as well as between the MDM Server and a trusted, remote administrator.

1.6.2.3 Identification and Authentication

The MDM Server requires that users, whether administrators or mobile device users (MD users), to be authenticated before any security related functionality is allowed. MD Users are required to enroll their device with the MDM Server using an MDM Agent.

Both the MDM Server and the MDM Agent are required to utilize X.509 certificates which includes the validation check of the certificate in relation with the TLS connection in order to create a secure

connection with the MDM Server and MDM Agent. This will also occur with connections between the MDM Server and administrators that have remote access via the web interface and the Audit Server.

1.6.2.4 Security Management

The MDM Server allows for two distinct roles: that of the Administrator and the mobile device user (MD User). The Administrator interacts with the MDM Server and MD User interacts with a device that hosts the MDM Agent. The MDM Server also supports user access to management functions by defining their role and the capabilities each role is allowed.

The MDM Server presents all functionality needed to manage its own security functions as well as to manage the mobile device policies that are transmitted to MDM Agents. This ensures that security management functions are only allowed use by authorized administrators. MD Users are only allowed necessary functions such as their enrolling in the MDM Server.

The MDM Agent holds functionality needed to create a secure connection with and enrollment in the MDM Server, implement any policies received from an appropriately enrolled MDM Server, and be able to account the results of applying such policies.

1.6.2.5 Protection of the TSF

The MDM Server and Agent act together in order to ensure all security related interactions between their components are protected from modification and disclosure by any undesirable entities.

The MDM Server contains the ability to perform self-tests to make sure that appropriate functionality is taking place and is maintained. The MDM Server is able to verify cryptographically during start-up that its executable image has not been tampered with or corrupted.

The MDM Server and MDM Agents utilize digital signatures to verify trusted updates which mitigate the risk of malicious changes to the TOE.

1.6.2.6 TOE Access

The MDM Server supports two GUI administrator interfaces that support two Administrator interfaces.

1.6.2.7 Trusted Path/Channels

The MDM Server uses TLS/HTTPS to create a secure communication channel between itself and remote administrators. It uses TLS to communicate with an Audit Server. The MDM Server also uses TLS to connect with MD Users over a protected channel via their MDM Agent on their mobile device.

1.7 TOE Documentation

The following document is included in the TOE's physical boundary:

- *Ivanti Endpoint Manager Mobile (EPMM) System 12 Common Criteria Administrative Guidance, Version 1.2, July 2025*.

1.8 Product Functionality not Included in the Scope of the Evaluation

- Ivanti Sentry is not include in the evaluation.

- The evaluated configuration does not include the CLI interface referred to as the Ivanti Self-Service User Portal user interface.
- The EPMM evaluation does not include the Non-admin and Guest roles supported by Android devices.
- The EPMM evaluated configuration does not support connecting to the MDM Server using the SSH network protocol. SSH, also known as Secure Shell or Secure Socket Shell, provides a CLI.
- The EPMM Agent that was evaluated is the Federal Build and is acquired through Ivanti personnel. It is not the Ivanti commercial EPMM Agent, also known as mobile@work, available via Google Play and the Ivanti website.
- The Trusted Front End (TFE) functionality is not included in the evaluated configuration.
- SAML is not enabled in the CC evaluated configuration.
- ActiveSync is not enabled in the evaluated configuration.

2 Conformance Claims

This section identifies the TOE conformance claims, conformance rationale, and relevant Technical Decisions (TDs).

2.1 CC Conformance Claims

The TOE is conformant to the following:

- *Common Criteria for Information Technology Security Evaluations Part 1*, Version 3.1, Revision 5, April 2017.
- *Common Criteria for Information Technology Security Evaluations Part 2*, Version 3.1, Revision 5, April 2017 (extended).
- *Common Criteria for Information Technology Security Evaluations Part 3*, Version 3.1, Revision 5, April 2017 (extended)**.**

2.2 Protection Profile Conformance

This ST and the TOE it describes are exact conformance to the following CC specifications:

- *PP-Configuration for Mobile Device Management (MDM) and MDM Agents*, Version 1.0, 27 January 2020 [CFG_MDM-MDM_AGENT_V1.0].

  This PP-Configuration includes the following components:

  - Base-PP: *Protection Profile for Mobile Device Management,* Version 4.0, 25 April 2019 (PP_MDM_V4.0).
  - PP-Module: *PP-Module for MDM Agents*, Version 1.0, 25 April 2019 (MOD_MDM_AGENT_V1.0).

Additionally, the ST claims exact conformance to the following functional packages:

- *Functional Package for Transport Layer Security (TLS)*, Version 1.1, 01 March 2019 [PKG_TLS_V1.1].

2.3 Conformance Rationale

This ST provides exact conformance to the items listed in the previous section. The security problem definition, security objectives, and security requirements in this ST are all taken from the PP_MDM_V4.0, MOD_MDM_AGENT_V1.0 and PKG_TLS_v1.1, performing only the operations defined there.

2.3.1 Technical Decisions

All NIAP Technical Decisions (TDs) issued to date and applicable to PP_MDM_V4.0, MOD_MDM_AGENT_V1.0, and PKG_TLS_v1.1 have been considered. The following table identifies all applicable TDs.

Table Key
MDM = Applicable to PP_MDM_V4.0.
AGT = Applicable to MOD_MDM_AGENT_V1.0.
TLS = Applicable to PKG_TLS_v1.1.

Applicable means the TD applies to the evaluation. Specifically, if a TD applies to an SFR claimed by the ST, the TD is applicable. The TD may or may not affect the evaluation (i.e. the TD modifies a selection the ST did not claim) however, the TD is still considered applicable in this ST.

Table 9: Relevant Technical Decisions (TDs)

Technical Decision	PP, Module, or Functional Package			Applicable (Yes/No)	Notes and Exclusion Rationale (if applicable)
	MDM	AGT	TLS		
PP_MDM_V4.0 and MOD_MDM_AGENT_V1.0 TDs					
TD0951 – Adding FIPS 186-5 in PP_MDM_V4.0	✓			Yes	Archives TD0872. Applies to FCS_CKM.1, FCS_COP.1(3) SFRs.
TD0935 – Clarification when CTR_DRBG is Selected for FCS_RBG_EXT.1.2 in PP_MDM_V4.0.	✓			Yes	Applies to FCS_RBG_EXT.1.2 SFR and App Note.
TD0922 – Clarification to Application Note for FCS_RBG_EXT.1.2.	✓			Yes	Applies to FCS_RBG_EXT.1.2 App Note.
TD0914 – Addition of PKG_TLS_2.0 To Conformance Claims	✓			Yes	Goes into effect 1 October 2025.
TD0895 – Third Party Libraries in FPT_LIB_EXT.1.1	✓			Yes	Applies to FPT_LIB_EXT.1.1 App Note.
TD0887 – Management of x509 certificates for cloud	✓			Yes	Applies to FMT_SMF.1(2) SFR.
TD0844 – Addition of Assurance Packate for Flaw Remediation V1.0 Conformance Claim	✓			No	Flaw remediation is not claimed for this evaluation.
TD0784 – Terminology Change in MDMPP: Extended Functional Package	✓			Yes	Applies to FIA_X509_EXT.2.1 SFR. Applies to FTP_ITC.1.1(1) SFR and App Note. Applies to FTP_TRP.1.1(1) SFR. Applies to FTP_TRP.1.3(1) App Note. Applies to FTP_ITT.1.1(1) SFR App Note. (not claimed)
TD0755 – MDM-Agent Policy Authenticity		✓		Yes	Archives TD0491. Applies to FMT_POL_EXT.2.1, FMT_POL_EXT.2.2, and FMT_SMF_EXT.4.1 SFRs. Applies to FCS_STG_EXT.4 App Note and TSS (not claimed).

Technical Decision	PP, Module, or Functional Package			Applicable (Yes/No)	Notes and Exclusion Rationale (if applicable)
	MDM	AGT	TLS		
					Applies to FMT_SMF_EXT.4 Test.
TD0754 – MDM Policy Authenticity	✓			Yes	Removes FIA_X509_EXT.5_EXT.1 X.509 Unique Certificate *(MDM_PP)* Adds FIA_CLI_EXT.1 Client Authorization *(MDM_PP)* SFR, AGD, and Test (claimed) Adds FIA_TOK_EXT.1 Client Tokens *(MDM_PP*) SFR, TSS, AGD, and Tests. (not claimed) Applies to FMT_POL_EXT.1 SFR and TSS.
TD0673 – MDM-Agent PP-Module updated to allow for new PP and PP-Module Versions		✓		No	MDF, biometrics, Bluetooth, or Wireless LAN are not part of evaluation.
TD0660 – Mislabeled SFRs in MDM Agent Auditable Events Table		✓		Yes	Applies to FAU_GEN.1(2) and FAU_SEL.1(2) entries in Auditable Events table.
TD0650 – Conformance claim sections updated to allow for MOD_VPNC_V2.3 and 2.4	✓	✓		No	VPN Client not applicable for this evaluation.
TD0641 – Alternative revocation checking for MDM	✓			Yes	Applies to FIA_X509.1.1(1) and FIA_X509_EXT.1.1(2) SFRs, TSS, AGD, and Test. Applies to FIA_X509_EXT.2.2 App Note.
TD0629 – Audit Events for Startup and Shutdown	✓			Yes	Applies to FAU_GEN.1(1) SFR and App Note.
TD0616 – MDM PP Use Case Mappings	✓			Yes	Applies only to the PP language in Appendix G.
TD0600 – Conformance claim sections updated to allow for MOD_VPNC_V2.3	✓	✓		No	MOD_VPNC_V2.3 is not part of the PP-config for this evaluation.
TD0594 – Distributed TOE tests in FCO_CPC_EXT.1.3	✓			Yes	Applies to FCO_CPC_EXT.1 Test.
TD0552 – SFR Rationale and Implicitly Satisfied SFRs	✓			Yes	Applies to the justification for each security objective for the TOE and implicitly satisfied requirements.
TD0497 – SFR Rationale, Consistency of SPD, and Implicitly Satisfied SFRs		✓		Yes	Applies to Objective to SFR rational,

Technical Decision	PP, Module, or Functional Package			Applicable (Yes/No)	Notes and Exclusion Rationale (if applicable)
	MDM	AGT	TLS		
					OSPs, Threats, Assumptions, Policies
TD0479 – FMT_SMF.1(1) Reliance on MDF Evals	✓			Yes	Applies to FMT_SMF.1(1) App Note and TSS.
TD0462 – MDM Distributed TOE: Registration Channel Updates	✓			Yes	Applies to FCO_CPC_EXT.1.2 SFR.
TD0461 – Security Audit for Distributed TOEs	✓			Yes	Clarity of audit for distributed TOEs.
TD0438 – TST and TUD on the MDM Agent	✓			Yes	Applies to FPT_TST_EXT.1 fulfillment of components and App Note. Applies to FPT_TUD_EXT.1 SFR and App Note. fulfillment of components.
PKG_TLS_V1.1					
TD0779 – Updated Session Resumption Support in TLS package V1.1			✓	Yes	TD0588 is archived. Applies to FCS_TLSS_EXT.1 SFR, App Note, and Test.
TD0770 – TLSS.2 connection with no client cert			✓	Yes	Applies to FCS_TLSS_EXT.2 SFR, App Note, TSS, and Test.
TD0739 – PKG_TLS_V1.1 has 2 different publication dates			✓	Yes	Applies to PKG_TLS_V1.1 publication date (01 March 2019). Applies to FCS_TLSS_EXT.1 Test.
TD0726 – Corrections to (D)TLSS SFRs in TLS 1.1 FP			✓	Yes	Applies to FCS_TLSS_EXT.1 SFR (claimed). Applies FCS_DTLSS_EXT.1 SFR (not claimed).
TD0513 – CA Certificate loading.			✓	Yes	Applies to FCS_TLSC_EXT.1 Test.
TD0499 – Testing with pinned certificates.			✓	Yes	Applies to FCS_TLSC_EXT.1 Test.
TD0469 – Modification of test activity for FCS_TLSS_EXT.1.1 test 4.1.			✓	Yes	Applies to FCS_TLSS_EXT.1 Test.
TD0442 – Updated TLS Ciphersuites for TLS Package			✓	Yes	Applies to FCS_TLSC_EXT.1.1 and FCS_TLSS_EXT.1.1 SFRs.

3 Security Problem Definition

The security problem definition has been taken directly from the claimed PP, Module, and Functional Package and is reproduced here for the convenience of the reader. The security problem is described in terms of the threats that the TOE is expected to address, assumptions about the operational environment, and any Organizational Security Policies (OSPs) that the TOE is expected to enforce.

3.1 Threats

The threats identified in the following table are drawn directly from PP_MDM_V4.0, MOD_MDM_AGENT_V1.0, and any TDs that apply to the evaluation. PKG_TLS_v1.1 does not define any threats. The threats listed below that apply only to the PP_MDM_V4.0 or MOD_MDM_AGENT_V1.0 are appended with /SRV or /AGT respectively.

Table 10: Threats

Threat ID	Threat
T.BACKUP/AGT	This threat protects user data from unauthorized logical access. If the backup data is stored outside the MDM or the mobile device that it protects, then there is no conflict with the MDM PP since it is a different security boundary. If the backup data is stored either on the MDM or on the protected device, an attacker would attempt to exploit this threat by first exploiting any of the threats that the MDM PP defines (T.MALICIOUS_APPS, T.NETWORK_ATTACK, T.NETWORK_EAVESDROP, T.PHYSICAL_ACCESS) depending on where and how the backup data is stored, and then use successful exploitation of one of these threats to attempt to access the backup data itself. *Addressed TD0497.*
T.MALICIOUS_APPS/SRV	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.NETWORK_ATTACK/SRV	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/SRV	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/SRV	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.

3.2 Assumptions

The assumptions identified in the following table are drawn directly from PP_MDM_V4.0, MOD_MDM_AGENT_V1.0, and any TDs that apply to the evaluation. PKG_TLS_v1.1 does not define any

assumptions. The Assumptions listed below that apply only to the PP_MDM_V4.0 or MOD_MDM_AGENT_V1.0 are appended with /SRV or /AGT respectively.

Table 11: Assumptions

Assumption ID	Assumption
A.COMPONENTS_RUNNING/SRV	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/AGT	This assumption expects that networking services will be available for the TOE to use. This is consistent with the Base-PP because the portion of the TOE defined by the Base-PP runs on a general-purpose operating system or specialized network appliance. The Base-PP does not make any assumptions about the environmental functionality that this PP-Module relies on, so there is nothing in this PP-Module that would contradict it. *Addressed TD0497.*
A.CONNECTIVITY/SRV	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.MOBILE_DEVICE_PLATFORM/AGT	This assumption expects that the TOE's underlying hardware platform is validated against the MDF PP. This is consistent with the Base-PP because the portion of the TOE defined by the Base-PP runs on a general-purpose operating system or specialized network appliance. The Base-PP does not make any assumptions about the environmental functionality that this PP-Module relies on, so there is nothing in this PP-Module that would contradict it. *Addressed TD0497.*
A.MDM_SERVER_PLATFORM/SRV	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.PROPER_ADMIN/AGT	The Base-PP defines an A.PROPER_ADMIN assumption that is identical to the one defined by the PP-Module. The PP-Module just extends it to the entire TOE boundary rather than just the MDM Server. *Addressed TD0497.*
A.PROPER_ADMIN/SRV	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.

Assumption ID	Assumption
A.PROPER_USER/AGT	The Base-PP defines an A.PROPER_USER assumption that is identical to the one defined by the PP-Module. The PP-Module just extends it to the entire TOE boundary rather than just the MDM Server. *Addressed TD0497.*
A.PROPER_USER/SRV	Mobile device users are not willfully negligent or hostile, and use the device within compliance of a reasonable Enterprise security policy.

3.3 Organizational Security Policies

The Organizational Security Policies (OSPs) identified in the following table are drawn directly from PP_MDM_V4.0, MOD_MDM_AGENT_V1.0, and any TDs that apply to the evaluation. PKG_TLS_v1.1 does not define any OSPs. The OSPs listed below that apply only to the PP_MDM_V4.0 or MOD_MDM_AGENT_V1.0 are appended with /SRV or /AGT respectively.

Table 12: Organizational Security Policies (OSPs) (*Applied TD0497*)

OSP ID	OSP
P.ACCOUNTABILITY/AGT	The Base-PP defines a P.ACCOUNTABILITY OSP that is identical to the one defined by the PP-Module. The PP-Module just extends it to the entire TOE boundary rather than just the MDM Server. *Addressed TD0497.*
P. .ACCOUNTABILITY/SRV	Personnel operating the TOE shall be accountable for their actions within the TOE.
P.ADMIN/AGT	The Base-PP defines a P.ADMIN OSP that is identical to the one defined by the PP-Module. The PP-Module just extends it to the entire TOE boundary rather than just the MDM Server. *Addressed TD0497.*
P.ADMIN/SRV	The configuration of the mobile device security functions must adhere to the Enterprise security policy.
P.DEVICE_ENROLL/AGT	The Base-PP defines a P.DEVICE_ENROLL OSP that is identical to the one defined by the PP-Module. The PP-Module just extends it to the entire TOE boundary rather than just the MDM Server. *Addressed TD0497.*
P.DEVICE_ENROLL/SRV	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/AGT	The Base-PP defines a P.NOTIFY OSP that is identical to the one defined by the PP-Module. The PP-Module just extends it to the entire TOE boundary rather than just the MDM Server. *Addressed TD0497.*
P.NOTIFY/SRV	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 Security Objectives

4.1 Security Objectives for the TOE

The security objectives for the TOE identified in the following table are drawn directly from PP_MDM_V4.0, MOD_MDM_AGENT_V1.0, and any TDs that apply to the evaluation. PKG_TLS_v1.1 does not define any security objectives for the TOE. The security objectives for the TOE listed below that apply only to the PP_MDM_V4.0 or MOD_MDM_AGENT_V1.0 are appended with /SRV or /AGT respectively.

Table 13: Security Objectives for the TOE (*Addressed TD0497*)

Security Objectives for the TOE ID	Security Objectives for the TOE
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/SRV	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/SRV	The TOE provides access controls around its management functionality.
O.QUALITY/SRV	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/AGT	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.

4.2 Security Objectives for the Operational Environment

The security objectives for the operational environment assist the TOE in correctly providing its security functionality. These objectives, which are found in the table below, track with the assumptions about the TOE operational environment. Security objectives for the operational environment identified in the following table are drawn directly from PP_MDM_V4.0, MOD_MDM_AGENT_V1.0, and any TDs that apply to the evaluation. PKG_TLS_v1.1 does not define any security objectives for the operational environment. The security objectives for the operational environment listed below that apply only to the PP_MDM_V4.0 or MOD_MDM_AGENT_V1.0 are appended with /SRV or /AGT respectively.

Table 14: Security Objectives for the Operational Environment

Security Objectives for the Operational Environment ID	Security Objectives for the Operational Environment
OE.COMPONENTS_RUNNING/SRV	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.DATA_PROPER_ADMIN/AGT	TOE Administrators are trusted to follow and apply all administrator guidance in a trusted manner.
OE.DATA_PROPER_USER/AGT	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/AGT	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.PROPER_ADMIN/SRV	TOE Administrators are trusted to follow and apply all administrator guidance in a trusted manner.
OE.PROPER_USER/SRV	Users of the mobile device are trained to securely use the mobile device and apply all guidance in a trusted manner.
OE.TIMESTAMP/SRV	Reliable timestamp is provided by the operational environment for the TOE.
OE.WIRELESS_NETWORK	A wireless network will be available to the mobile devices.

5 Security Requirements

This section identifies the Security Functional Requirements (SFRs) and the Security Assurance Requirements (SARs) for the TOE. The Security Requirements included in this section are derived from Part 2 of the *Common Criteria for Information Technology Security Evaluation*, Version 3.1, Revision 5, April 2017; PP_MDM_V4.0, MOD_MDM_AGENT_V1.0, and PKG_TLS_v1.1.

5.1 Conventions

This section includes the definition of naming of the SFRs including the operations. The CC allows the following types of operations to be performed on the functional requirements: assignments, selections, refinements, and iterations. The following font conventions are used within this document to identify operations defined by CC and are applied only to the operations that are available by the PP:

- Assignment: Indicated with *italicized* text;
- Refinement: Indicated with **bold** text;
- Selection: Indicated with underlined text;
- Iteration:
  - The PP and Module authors indicate iterations by an SFR, followed by a number enclosed in parenthesis, e.g., FTP_TRP.1(1).
  - The ST author indicates iterations of /SRV and /AGT if an SFR name is repeated in the documents. The /SRV and /AGT indicate the SFR is from the PP_MDM_V4.0 PP or the MOD_MDM_AGENT_V1.0 Module respectively.
- Formatting used in the PP/Module/Functional Spec is retained except for text within brackets. If the text is an operation completed by the ST author, the formatting described in the first three bullet items of this section are applied. Otherwise, the text within brackets is plaintext.
- Section 2.3.1 identifies the TDs that are applicable to this evaluation and defines what applicable means in this ST. If a TD identifies an SFR and is applicable to the evaluation, the SFR will follow with an Application Note indicating the TD was addressed. The TD may or may not affect the evaluation (i.e. the TD modifies a selection the ST did not claim), the TD still is marked as Addressed.

5.2 Extended Functional Requirements

All the extended requirements in this ST have been drawn from PP_MDM_V4.0, MOD_MDM_AGENT_V1.0, PKG_TLS_v1.1, and any TDs. These documents define the extended SFRs. Since they have not been redefined in this ST, the PP_MDM_V4.0, MOD_MDM_AGENT_V1.0, and PKG_TLS_v1.1 should be consulted for more information regarding these extensions to CC Part 2.

Extended SFRs from PP_MDM_V4.0:

- FAU_ALT_EXT.1
- FAU_CRP_EXT.1
- FAU_NET_EXT.1
- FAU_STG_EXT.1
- FAU_STG_EXT.2
- FCO_CPC_EXT.1
- FCS_CKM_EXT.4
- FCS_HTTPS_EXT.1
- FCS_RBG_EXT.1

- FCS_STG_EXT.1
- FCS_STG_EXT.2
- FIA_CLI_EXT.1
- FIA_ENR_EXT.1
- FIA_X509_EXT.1(1)
- FIA_X509_EXT.2
- FMT_POL_EXT.1
- FPT_API_EXT.1
- FPT_LIB_EXT.1
- FPT_TST_EXT.1
- FPT_TUD_EXT.1
- FTP_ITC_EXT.1

Extended SFRs from MOD_MDM_AGENT_V1.0:

- FAU_ALT_EXT.2
- FCS_STG_EXT.1(2)
- FIA_ENR_EXT.2
- FMT_POL_EXT.2
- FMT_SMF_EXT.4
- FMT_UNR_EXT.1

Extended SFRs from PKG_TLS_v1.1:

- FCS_TLS_EXT.1
- FCS_TLSC_EXT.1
- FCS_TLSC_EXT.2
- FCS_TLSC_EXT.5
- FCS_TLSS_EXT.1
- FCS_TLSS_EXT.2
- FCS_TLSS_EXT.4

5.3 Security Functional Requirements List

Table 15: Security Functional Requirements (SFRs)

Family	Requirement	Description
Security audit (FAU)	FAU_ALT_EXT.1	Server Alerts *(MDM_PP)*
	FAU_ALT_EXT.2	Agent Alerts *(AGT_PPM)*
	FAU_CRP_EXT.1	Support for Compliance Reporting of Mobile Device Configuration *(MDM_PP)*
	FAU_GEN.1(1)	Audit Data Generation *(MDM_PP)*
	FAU_GEN.1(2)/AGT	Audit Data Generation *(AGT_PPM)*
	FAU_GEN.1(2)/SRV	Audit Generation (MAS Server) *(MDM_PP)*
	FAU_NET_EXT.1	Network Reachability Review *(MDM_PP)*
	FAU_SAR.1	Audit Review *(MDM_PP)*

Family	Requirement	Description
	FAU_SEL.1(2)	Security Audit Event Selection *(AGT_PPM)*
	FAU_STG_EXT.1	External Trail Storage *(MDM_PP)*
	FAU_STG_EXT.2	Audit Event Storage *(MDM_PP)*
Communication (FCO)	FCO_CPC_EXT.1	Component Registration Channel Definition *(MDM_PP)*
Cryptographic Support (FCS)	FCS_CKM.1	Cryptographic Key Generation *(MDM_PP)*
	FCS_CKM.2	Cryptographic Key Establishment *(MDM_PP)*
	FCS_CKM_EXT.4	Cryptographic Key Destruction *(MDM_PP)*
	FCS_COP.1.1(1)	Cryptographic Operation (Confidentiality Algorithms) *(MDM_PP)*
	FCS_COP.1.1(2)	Cryptographic Operation (Hashing Algorithms) *(MDM_PP)*
	FCS_COP.1.1(3)	Cryptographic Operation (Signature Algorithms) *(MDM_PP)*
	FCS_COP.1.1(4)	Cryptographic Operation (Keyed-Hash Message Authentication) *(MDM_PP)*
	FCS_HTTPS_EXT.1	HTTPS Protocol `*(MDM_PP)*
	FCS_IV_EXT.1	Initialization Vector Generation *(MDM_PP)*
	FCS_RBG_EXT.1/AGT	Extended: Random Bit Generation *(MDM_PP)*
	FCS_RBG_EXT.1/SRV	Extended: Random Bit Generation *(MDM_PP)*
	FCS_STG_EXT.1	Cryptographic Key Storage *(MDM_PP)*
	FCS_STG_EXT.1(2)	Cryptographic Key Storage *(AGT_PPM)*
	FAU_STG_EXT.2	Audit Event Storage *(MDM_PP)*
	FCS_TLS_EXT.1	TLS Protocol *(TLS_PKG)*
	FCS_TLSC_EXT.1	TLS Client Protocol *(TLS_PKG)*
	FCS_TLSC_EXT.2	TLS Client Support for Mutual Authentication *(TLS_PKG)*
	FCS_TLSC_EXT.5	TLS Client Support for Supported Groups Extension *(TLS_PKG)*
	FCS_TLSS_EXT.1	TLS Server Protocol *(TLS_PKG)*
	FCS_TLSS_EXT.2	TLS Server Support for Mutual Authentication *(TLS_PKG)*
Identification and authentication (FIA)	FIA_CLI_EXT.1	X.509 Unique Certificate *(MDM_PP)*
	FIA_ENR_EXT.1	Enrollment of Mobile Device into Management *(MDM_PP)*
	FIA_ENR_EXT.2	Agent Enrollment of Mobile Device into Management *(AGT_PPM)*
	FIA_UAU.1	Timing of Authentication *(MDM_PP)*
	FIA_X509_EXT.1(1)	X.509 Certification Validation *(MDM_PP)*
	FIA_X509_EXT.2	X.509 Certificate Authentication *(MDM_PP)*
Security management (FMT)	FMT_MOF.1(1)	Management of Functions Behaviour *(MDM_PP)*
	FMT_MOF.1(2)	Management of functions Behaviour (Enrollment) *(MDM_PP)*
	FMT_MOF.1(3)	Management of Functions in (MAS Server Downloads) *(MDM_PP)*

Family	Requirement	Description
	FMT_POL_EXT.1	Trusted Policy Update *(MDM_PP)*
	FMT_POL_EXT.2	Agent Trusted Policy Update *(AGT_PPM)*
	FMT_SMF.1(1)	Specification of Management Functions (Server configuration of Agent) *(MDM_PP)*
	FMT_SMF.1(2)	Specification of Management Functions (Server Configuration of Server) *(MDM_PP)*
	FMT_SMF.1(3)	Specification of Management Functions (MAS Server) *(MDM_PP)*
	FMT_SMF_EXT.4	Specification of management Functions *(AGT_PPM)*
	FMT_SMR.1(1)	Security Management Roles *(MDM_PP)*
	FMT_SMR.1(2)	Security Management Roles (MAS Server) *(MDM_PP)*
	FMT_UNR_EXT.1	User Unenrollment Prevention *(AGT_PPM)*
Protection of the TSF (FPT)	FPT_API_EXT.1	Use of Supported Services and API's *(MDM_PP)*
	FPT_ITT.1(2)	Internal TOE TSF Data Transfer (MDM Agent) *(MDM_PP)*
	FPT_LIB_EXT.1/AGT	Use of Third Party Libraries *(MDM_PP)*
	FPT_LIB_EXT.1/SRV	Use of Third Party Libraries *(MDM_PP)*
	FPT_TST_EXT.1	Functionality Testing *(MDM_PP)*
	FPT_TUD_EXT.1	Trusted Update *(MDM_PP)*
Trusted path/channels (FTP)	FTP_ITC.1(1)	Inter-TSF Trusted Channel (Authorized IT Entities) *(MDM_PP)*
	FTP_ITC_EXT.1	Trusted Channel *(MDM_PP)*
	FTP_TRP.1(1)	Trusted Path (for Remote Administration) *(MDM_PP)*
	FTP_TRP.1(2)	Trusted Path (for Enrollment) *(MDM_PP)*

5.4 Security Functional Requirements (SFRs)

This section includes the security functional requirements (SFR) claimed by this ST. Following each SFR is the identity of the PP, Module, or Package that sourced the SFR.

- *(MDM_PP)* indicates the SFR is from the *Protection Profile for Mobile Device Management,* version 4.0 PP*;*
- *(AGT_PPM)* indicates the SFR is from the *PP-Module for MDM Agent,* version 1.0 PP Module; and
- *(TLS_PKG)* indicates the SFR is from the *Functional Package for Transport Layer Security (TLS),* version 1.1 Functional Package.

5.4.1 Security Audit (FAU)

5.4.1.1 FAU_ALT_EXT.1 Server Alerts *(MDM_PP)*

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. [no other events].

5.4.1.2 FAU_ALT_EXT.2 Agent Alerts *(AGT_PPM)*

FAU_ALT_EXT.2.1 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,
- [
  - no other events,

].

FAU_ALT_EXT.2.2 The MDM Agent shall queue alerts if the trusted channel is not available.

5.4.1.3 FAU_CRP_EXT.1 Support for Compliance Reporting of Mobile Device Configuration *(MDM_PP)*

FAU_CRP_EXT.1.1 The TSF shall provide [

- an interface that provides responses to queries about the configuration of enrolled devices,
- an interface that permits the export of data about the configuration of enrolled devices

] to authorized entities over a channel that meets the secure channel requirements in FTP_ITC.1(1). The provided information for each enrolled mobile device includes:

a. The current version of the MD firmware/software
b. The current version of the hardware model of the device
c. The current version of installed mobile applications
d. List of MD configuration policies that are in place on the device (as defined in FMT_SMF.1.1(1))
e. [no other information]

Application Note: FTP_ITC.1(1) Inter-TSF Trusted Channel (Authorized IT Entities) would be appropriate for a non-distributed TOE. However, this ST satisfies the connection with FPT_ITT.1(2) Internal TOE TSF Data Transfer (MDM Agent) (MDM_PP).

5.4.1.4 FAU_GEN.1(1) Audit Data Generation *(MDM_PP)*

FAU_GEN.1.1(1) **Refinement:** The TSF shall [implement functionality] to generate an audit record of the following auditable events:

a) All administrative actions
b) [none]
c) Specifically defined auditable events listed in **Table 16.**
d) [no other events].

Refinement Rationale: The table reference is changed to reflect the contents of the ST.
Application Note: Addressed TD0629.

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 16.**
- [*no other audit relevant information*].

Refinement Rationale: The table reference is changed to reflect the contents of the ST.

Table 16: MDM Server Security Functional Requirements and Auditable Events

Requirement	Auditable Events	Additional Audit Record Contents
Security audit (FAU)		
FAU_ALT_EXT.1	Type of alert.	Identity of Mobile Device that sent alert.
FAU_CRP_EXT.1	None.	----------
FAU_GEN.1(1)	None.	----------
FAU_GEN.1(2)/SRV	None.	----------
FAU_NET_EXT.1	None.	----------
FAU_SAR.1	None.	----------
FAU_STG_EXT.1	None.	---------
FAU_STG_EXT.2	None.	---------
Communication (FCO)		
FCO_CPC_EXT.1	Enabling or disabling communications between a pair of components	Identities of the endpoints pairs enabled or disabled.
Cryptographic Support (FCS)		
FCS_CKM.1	[None].	No additional information.
FCS_CKM.2	None.	----------
FCS_CKM_EXT.4	None.	----------
FCS_COP.1(1)	None.	----------
FCS_COP.1(2)	None.	----------
FCS_COP.1(3)	None.	----------
FCS_COP.1(4)	None.	----------
FCS_HTTPS_EXT.1	Failure of the certificate validity check.	Issuer Name and Subject Name of certificate. [no additional information]
FCS_IV_EXT.1	None.	----------
FCS_RBG_EXT.1/SRV	Failure of the randomization process.	No additional information.
FCS_STG_EXT.1	None.	----------
FCS_STG_EXT.2	None.	----------
FCS_TLS_EXT.1	None.	----------
FCS_TLSC_EXT.1	Failure to establish a TLS session.	Reason for failure.
	Failure to verify presented identifier.	Presented identifier and reference identifier.
FCS_TLSC_EXT.2	None.	----------

Requirement	Auditable Events	Additional Audit Record Contents
FCS_TLSC_EXT.5	None.	----------
FCS_TLSS_EXT.1	Failure to establish a TLS session.	Reason for failure.
FCS_TLSS_EXT.2	None.	----------
Identification and authentication (FIA)		
FIA_CLI_EXT.1[1]	None.	----------
FIA_ENR_EXT.1	Failure of MD user authentication.	Presented username.
FIA_UAU.1	None.	---------
FIA_X509_EXT.1(1)	Failure to validate X.509 certificate.	Reason for failure.
FIA_X509_EXT.2	Failure to establish connection to determine revocation status.	No additional information.
Security management (FMT)		
FMT_MOF.1(1)	Issuance of command to perform function.	Command sent and identity of MDM Agent recipient(s).
	Change of policy settings.	Policy changed and value or full policy.
FMT_MOF.1(2)	Enrollment by a user.	Identity of user
FMT_MOF.1(3)	None.	----------
FMT_POL_EXT.1	None.	----------
FMT_SMF.1(1)	None.	----------
FMT_SMF.1(2)	Success or failure of function.	No additional information.
FMT_SMF.1(3)	None.	----------
FMT_SMR.1(1)	None.	---------
FMT_SMR.1(2)	None.	----------
Protection of the TSF (FPT)		
FPT_API_EXT.1	None.	----------
FPT_ITT.1(2)	Initiation and termination of the trusted channel.	Trusted channel protocol. Identity of initiator and recipient.
FPT_LIB_EXT.1	None.	----------
FPT_TST_EXT.1	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	Success or failure of signature verification.	No additional information
Trusted path/channels (FTP)		
FTP_ITC.1(1)	Initiation and termination of the trusted channel.	Trusted channel protocol. Non-TOE endpoint of connection.

[1] TD0754 replaced FCS_TLSC_EXT.5 X.509 Unique Certificate with FIA_CLI_EXT.1 Client Authorization. The TD did not specify the audit requirements and therefore, the ST author chose to use the FCS_TLS_EXT.5 requirements.

Requirement	Auditable Events	Additional Audit Record Contents
FTP_ITC_EXT.1	None.	----------
FTP_TRP.1(1)	Initiation and termination of the trusted channel.	Trusted channel protocol. Identity of administrator.
FTP_TRP.1(2)	Initiation and termination of the trusted channel.	Trusted channel protocol.

5.4.1.5 FAU_GEN.1(2)/AGT Audit Data Generation *(AGT_PPM)*

FAU_GEN.1.1(2)/AGT **Refinement:** The MDM Agent shall [implement functionality] to generate an MDM Agent audit record of the following auditable events:

a. Startup 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 17, and [no other events]

].

Refinement Rationale: The table reference is changed to reflect the contents of the ST.

FAU_GEN.1.2(2)/AGT **Refinement:** 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, and additional information in Table 17; 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*].

Refinement Rationale: The table reference is changed to reflect the contents of the ST.
Application Note: This instance of FAU_GEN.1(2) applies to the MDM Agent because it is included in MOD_MDM_AGENT_V1.0 and the FAU_GEN.1.2(2) instance listed below is identified in PP_MDM_V4.0 applies to the MDM Server. The SFR iteration conventions used by the ST author are identified in Section 5.1.
Application Note: Addressed TD0660.

Table 17: EPMM Agent *(AGT_PPM)* Security Functional Requirements and Auditable Events (addressed TD0660)

Requirement	SFR Source PP/Module /Functional Package	Auditable Events	Additional Audit Record Contents
Security audit (FAU)			
FAU_ALT_EXT.2	MOD_MDM_AGENT_V1.0	Success/failure of sending alert.	No additional information.
FAU_GEN.1(2)	MOD_MDM_AGENT_V1.0	None.	N/A

Requirement	SFR Source PP/Module /Functional Package	Auditable Events	Additional Audit Record Contents
FAU_SEL.1(2)	MOD_MDM_AGENT_V1.0	All modifications to the audit configuration that occur while the audit collection functions are operating.	No additional information.
FAU_STG_EXT.1	PP_MDM_V4.0	None.	----------
FAU_STG_EXT.2	PP_MDM_V4.0	None.	----------
Cryptographic Support (FCS)			
FCS_CKM.1	PP_MDM_V4.0	[None].	No additional information.
FCS_CKM.2	PP_MDM_V4.0	None.	----------
FCS_CKM_EXT.4	PP_MDM_V4.0	None.	----------
FCS_COP.1.1(1)	PP_MDM_V4.0	None.	----------
FCS_COP.1.1(2)	PP_MDM_V4.0	None.	----------
FCS_COP.1.1(3)	PP_MDM_V4.0	None.	----------
FCS_COP.1.1(4)	PP_MDM_V4.0	None.	----------
FCS_HTTPS_EXT.1	PP_MDM_V4.0	Failure of the certificate validity check.	Issuer Name and Subject Name of certificate. [no additional information]
FCS_RBG_EXT.1/AGT	PP_MDM_V4.0	Failure of the randomization process.	No additional information.
FCS_STG_EXT.1(2)	MOD_MDM_AGENT_V1.0	None.	----------
FCS_TLS_EXT.1	PKG_TLS_v1.1	None.	----------
FCS_TLSC_EXT.1	PKG_TLS_v1.1	Failure to establish a TLS session.	Reason for failure.
		Failure to verify presented identifier.	Presented identifier and reference identifier.
		Establishment/termination of a TLS session	Non-TOE endpoint of connection.

Requirement	SFR Source PP/Module /Functional Package	Auditable Events	Additional Audit Record Contents
FCS_TLSC_EXT.2	PKG_TLS_v1.1	None.	----------
FCS_TLSC_EXT.5	PKG_TLS_v1.1	None.	----------
Identification and authentication (FIA)			
FIA_ENR_EXT.2	MOD_MDM_AGENT_V1.0	Enrollment in management.	Reference identifier of MDM Server.
FIA_X509_EXT.1(1)	PP_MDM_V4.0	Failure to validate X.509 certificate.	Reason for failure.
FIA_X509_EXT.2	PP_MDM_V4.0	Failure to establish connection to determine revocation status.	No additional information.
Security management (FMT)			
FMT_POL_EXT.2	MOD_MDM_AGENT_V1.0	Failure of policy validation.	Reason for failure of validation.
FMT_SMF_EXT.4	MOD_MDM_AGENT_V1.0	Outcome (Success/failure) of function.	No additional information.
FMT_UNR_EXT.1	MOD_MDM_AGENT_V1.0	[Attempt to unenroll]	No additional information.
Protection of the TSF (FPT)			
FPT_API_EXT.1	PP_MDM_V4.0	None.	----------
FPT_LIB_EXT.1	PP_MDM_V4.0	None.	----------
FPT_TUD_EXT.1	PP_MDM_V4.0	Success or failure of signature verification.	No additional information

5.4.1.6 FAU_GEN.1(2)/SRV Audit Generation (MAS Server) *(MDM_PP)*

FAU_GEN.1.1(2)/SRV **Refinement**: 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)/SRV **Refinement:** 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 audit relevant information*].

Application Note: This instance of FAU_GEN.1(2) applies to the MDM Server because it is included in PP_MDM_V4.0 and the FAU_GEN.1.2(2) instance listed before is identified in MOD_MDM_AGENT_V1.0 applies to the MDM Agent. The SFR iteration conventions used by the ST author are identified in Section 5.1.

5.4.1.7 FAU_NET_EXT.1 Network Reachability Review *(MDM_PP)*

FAU_NET_EXT.1.1 The TSF shall provide authorized administrators with the capability to read the network connectivity status of an enrolled agent.

5.4.1.8 FAU_SAR.1 Audit Review *(MDM_PP)*

FAU_SAR.1.1 **Refinement**: The TSF shall [implement functionality] to provide [Authorized Administrators] with the capability to read [all audit data] from the audit records.

FAU_SAR.1.2 Refinement: The TSF shall [implement functionality] to provide the audit records in a manner suitable for the Authorized Administrators to interpret the information.

5.4.1.9 FAU_SEL.1(2) Security Audit Event Selection *(AGT_PPM)*

FAU_SEL.1.1(2) **Refinement:** The TSF shall [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*]

].

5.4.1.10 FAU_STG_EXT.1 External Trail Storage *(MDM_PP)*

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 [store audit data locally].

5.4.1.11 FAU_STG_EXT.2 Audit Event Storage *(MDM_PP)*

FAU_STG_EXT.2.1 The TSF shall [implement functionality] to protect the stored audit records in the audit trail from unauthorized modification.

5.4.2 Communication (FCO)

5.4.2.1 FCO_CPC_EXT.1 Component Registration Channel Definition*(MDM_PP)*

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 The TSF shall [implement 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 [FPT_ITT.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 the TOE components.

Application Note: Addressed TD0462.

5.4.3 Cryptographic Support (FCS)

5.4.3.1 FCS_CKM.1 Cryptographic Key Generation *(MDM_PP)*

FCS_CKM.1.1 **Refinement:** The TSF shall [implement 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 or FIPS PUB 186-5, "Digital Signature Standard (DSS)", Appendix A,
- ECC schemes using "NIST curves" P-384 and [P-256, P-521] that meet the following: FIPS PUB 186-4, "Digital Signature Standard (DSS)", *Appendix B.4 or* FIPS PUB 186-5, "Digital Signature Standard (DSS)", Appendix A.2,

].

Application Note: Addressed TD0951.

5.4.3.2 FCS_CKM.2 Cryptographic Key Establishment *(MDM_PP)*

FCS_CKM.2.1 **Refinement:** The TSF shall [implement functionality] **to perform** cryptographic **key establishment** in accordance with a specified cryptographic key **establishment** method: [

- RSA-based key establishment schemes that meet 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",
- Elliptic curve-based key establishment schemes that meet the following: NIST Special Publication 800-56A, “Recommendation for Pair-Wise Key Establishment Schemes Using Discrete Logarithm Cryptography”;

].

5.4.3.3 FCS_CKM_EXT.4 Cryptographic Key Destruction *(MDM_PP)*

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 [zeroes]]

- implementing key destruction in accordance with the following rules:
  - For volatile memory, the destruction shall be executed by a single direct overwrite [consisting of zeroes]
  - For non-volatile EEPROM, the destruction shall be executed by a single direct overwrite consisting of a pseudo-random pattern using the TSF/Platform RBG (as specified in FCS_RBG_EXT.1), followed by a read-verify.
  - For non-volatile flash memory, that is not wear-leveled, the destruction shall be executed [by a single direct overwrite consisting of zeros followed by a read-verify]
  - For non-volatile flash memory, that is wear-leveled, the destruction shall be executed [by a single direct overwrite consisting of zeros]
  - For non-volatile memory other than EEPROM and flash, the destruction shall be executed by a single direct overwrite with a random pattern that is changed before each write

].

FCS_CKM_EXT.4.2 The TSF shall destroy all plaintext keying material and critical security parameters (CSPs) when no longer needed.

5.4.3.4 FCS_COP.1(1) Cryptographic Operation (Confidentiality Algorithms) *(MDM_PP)*

FCS_COP.1.1(1) **Refinement:** The TSF shall [implement functionality] to perform encryption/decryption in accordance with a specified cryptographic algorithm: [

- AES-GCM (as defined in NIST SP 800-38D)

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

5.4.3.5 FCS_COP.1(2) Cryptographic Operation (Hashing Algorithms) *(MDM_PP)*

FCS_COP.1.1(2) **Refinement:** The TSF shall [implement 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.

5.4.3.6 FCS_COP.1(3) Cryptographic Operation (Signature Algorithms) *(MDM_PP)*

FCS_COP.1.1(3) **Refinement:** The TSF shall [implement 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 or FIPS PUB 186-5, '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 or FIPS PUB 186-5, 'Digital Signature Standard (DSS)', Section 5

].

Application Note: Applied TD0951.

5.4.3.7 FCS_COP.1(4) Cryptographic Operation (Keyed-Hash Message Authentication) *(MDM_PP)*

FCS_COP.1.1(4) **Refinement:** The TSF shall [implement functionality] to perform keyed-hash message authentication in accordance with a specified cryptographic algorithm HMAC -[SHA-256, SHA-384], key sizes [*256 and 384 bits used in HMAC*], 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.".

5.4.3.8 FCS_HTTPS_EXT.1 HTTPS Protocol *(MDM_PP)*

FCS_HTTPS_EXT.1.1 The TSF shall implement HTTPS using protocol that complies with RFC 2818.

FCS_HTTPS_EXT.1.2 The TSF shall implement HTTPS using TLS in accordance with the Package for Transport Layer Security.

5.4.3.9 FCS_IV_EXT.1 Initialization Vector Generation *(MDM_PP)*

FCS_IV_EXT.1.1 The TSF shall [implement functionality] to generate IVs in accordance with **Table 17.**

Refinement Rationale: The table reference is changed to reflect the contents of the ST.

Table 18: Reference and IV Requirements for NIST-approved Cipher Modes

Cipher Mode	Reference	IV Requirement
Galois Counter Mode (GCM)	SP800- 38D	IV shall be non-repeating. The number of invocations of GCM shall not exceed 2^32 for a given secret key unless an implementation only uses 96-bit IVs (default length).

5.4.3.10 FCS_RBG_EXT.1/AGT Random Bit Generation *(MDM_PP)*

FCS_RBG_EXT.1.1/AGT The TSF shall [implement 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/AGT The deterministic RBG shall be seeded by an entropy source that accumulates entropy from [a platform-based RBG, a hardware-based noise source] with a

minimum of [256 bits] of entropy at least equal to the greatest security strength (according to NIST SP 800-57) of the keys and hashes that it will generate.

Application Note: Addressed TD0935.

5.4.3.11 FCS_RBG_EXT.1/SRV Random Bit Generation *(MDM_PP)*

FCS_RBG_EXT.1.1/SRV The TSF shall [implement functionality] to perform all deterministic random bit generation services in accordance with NIST Special Publication 800-90A using [Hash_DRBG (any), CTR_DRBG(AES)] .

FCS_RBG_EXT.1.2/SRV The deterministic RBG shall be seeded by an entropy source that accumulates entropy from [

- a platform-based RBG,
- a hardware-based noise source

] with a minimum of [256 bits] of entropy at least equal to the greatest security strength (according to NIST SP 800-57) of the keys and hashes that it will generate.

Application Note: Addressed TD0935.

5.4.3.12 FCS_STG_EXT.1 Cryptographic Key Storage *(MDM_PP)*

FCS_STG_EXT.1.1 The TSF shall utilize [encryption as specified in FCS_STG_EXT.2] for all persistent secrets and private keys.

Application Note: This SFR applies only to the MDM Server. The following SFR, FCS_STG_EXT.1(2) Cryptographic Key Storage (AGT_PPM) is mandatory and covers key storage for the MDM Agents.

5.4.3.13 FCS_STG_EXT.1(2) Cryptographic Key Storage *(AGT_PPM)*

FCS_STG_EXT.1.1(2) **Refinement:** The MDM Agent shall use the [platform-provided key storage] for all persistent secret and private keys.

5.4.3.14 FCS_STG_EXT.2 Encrypted Cryptographic Key Storage *(MDM_PP)*

FCS_STG_EXT.2.1 The TSF shall [implement functionality] to encrypt all keys using AES in the [GCM mode] .

5.4.3.15 FCS_TLS_EXT.1 TLS Protocol *(TLS_PKG)*

FCS_TLS_EXT.1.1 **The product shall implement [**

- TLS as a client
- TLS as a server

].

5.4.3.16 FCS_TLSC_EXT.1 TLS Client Protocol *(TLS_PKG)*

FCS_TLSC_EXT.1.1 The product shall implement TLS 1.2 (RFC 5246) and [no earlier TLS versions] as a client that supports the cipher suites [

- TLS_RSA_WITH_AES_128_GCM_SHA256 as defined in RFC 5288,
- TLS_RSA_WITH_AES_256_GCM_SHA384 as defined in RFC 5288,

- TLS_ECDHE_ECDSA_WITH_AES_128_GCM_SHA256 as defined in RFC 5289,
- TLS_ECDHE_ECDSA_WITH_AES_256_GCM_SHA384 as defined in RFC 5289,
- TLS_ECDHE_RSA_WITH_AES_128_GCM_SHA256 as defined in RFC 5289,
- TLS_ECDHE_RSA_WITH_AES_256_GCM_SHA384 as defined in RFC 5289,

] and also supports functionality for [

- mutual authentication,

].

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

FCS_TLSC_EXT.1.3 The product shall not establish a trusted channel if the server certificate is invalid [

- with no exceptions,

].

Application Note: Addressed TD0442.

5.4.3.17 FCS_TLSC_EXT.2 TLS Client Support for Mutual Authentication *(TLS_PKG)*

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

5.4.3.18 FCS_TLSC_EXT.5 TLS Client Support for Supported Groups Extension *(TLS_PKG)*

FCS_TLSC_EXT.5.1 The product shall present the Supported Groups Extension in the Client Hello with the supported groups [secp256r1, secp384r1, secp521r1].support mutual authentication using X.509v3 certificates.

5.4.3.19 FCS_TLSS_EXT.1 TLS Server Protocol *(TLS_PKG)*

FCS_TLSS_EXT.1.1 The product shall implement TLS 1.2 (RFC 5246) and [no earlier TLS versions] as a server that supports the cipher suites [

- TLS_RSA_WITH_AES_128_GCM_SHA256 as defined in RFC 5288,
- TLS_RSA_WITH_AES_256_GCM_SHA384 as defined in RFC 5288,
- TLS_ECDHE_ECDSA_WITH_AES_128_GCM_SHA256 as defined in RFC 5289,
- TLS_ECDHE_ECDSA_WITH_AES_256_GCM_SHA384 as defined in RFC 5289,
- TLS_ECDHE_RSA_WITH_AES_128_GCM_SHA256 as defined in RFC 5289,
- TLS_ECDHE_RSA_WITH_AES_256_GCM_SHA384 as defined in RFC 5289

] and also supports functionality for [

- session resumption based on session IDs according to RFC 4346 (TLS1.1) or RFC 5246 (TLS1.2),
- session resumption based on session tickets according to RFC 5077

], and [

- mutual authentication,
- session renegotiation

].

FCS_TLSS_EXT.1.2 The product shall deny connections from clients requesting SSL 2.0, SSL 3.0, TLS 1.0 and [TLS 1.1].

FCS_TLSS_EXT.1.3 The product shall perform key establishment for TLS using [

- RSA with size [2048 bits, 3072 bits, 4096 bits],
- ECDHE parameters using elliptic curves [secp256r1, secp384r1, secp521r1] and no other curves

].

Application Note: Addressed TD0442, TD0726, and TD0779.

5.4.3.20 FCS_TLSS_EXT.2 TLS Server Support for Mutual Authentication *(TLS_PKG)*

FCS_TLSS_EXT.2.1 The product shall support authentication of TLS clients using X.509v3 certificates.

FCS_TLSS_EXT.2.2 The product shall [not establish a trusted channel] if the client certificate is invalid.

FCS_TLSS_EXT.2.3 The product shall not establish a trusted channel if the Distinguished Name (DN) or Subject Alternative Name (SAN) contained in a certificate does not match one of the expected identifiers for the client.

Application Note: Addressed by TD0770.

5.4.4 Identification and Authentication (FIA)

5.4.4.1 FIA_CLI_EXT.1 Client Authorization *(MDM_PP)*

FIA_CLI_EXT.1.1 The TSF shall require a unique [certificate] for each client device.

Application Note: This Mandatory SFR was added by TD0754 (the TD also deleted FIA_X509_EXT.5).

5.4.4.2 FIA_ENR_EXT.1 Enrollment of Mobile Device into Management *(MDM_PP)*

FIA_ENR_EXT.1.1 The TSF shall authenticate the remote users over a trusted channel during the enrollment of a mobile device.

FIA_ENR_EXT.1.2 The TSF shall limit the user's enrollment of devices to devices specified by [*[user credentials]*] and [a number of devices].

5.4.4.3 FIA_ENR_EXT.2 Agent Enrollment of Mobile Device into Management *(AGT_PPM)*

FIA_ENR_EXT.2.1 The MDM Agent shall record the reference identifier of the MDM Server during the enrollment process.

5.4.4.4 FIA_UAU.1 Timing of Authentication *(MDM_PP)*

FIA_UAU.1.1 **Refinement:** The TSF shall [implement functionality] to allow [*no TSF mediated actions*] on behalf of the user to be performed before the user is authenticated with the Server.

FIA_UAU.1.2 **Refinement:** 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.

5.4.4.5 FIA_X509_EXT.1(1) X.509 Certificate Validation *(MDM_PP)*

FIA_X509_EXT.1.1(1) The TSF shall [implement functionality] to validate certificates in accordance with the following rules:

- RFC 5280 certificate validation and certificate path validation
- The certificate path must terminate with a trusted CA certificate
- The TSF shall validate a certificate path by ensuring the presence of the basicConstraints extension, that the CA flag is set to TRUE for all CA certificates, and that any path constraints are met.
- The TSF shall validate that any CA certificate includes caSigning purpose in the key usage field
- The TSF shall validate the revocation status of the certificate using [
  - the Online Certificate Status Protocol (OCSP) as specified in RFC 2560,
  - CRL as specified in RFC 5759 Section 5

  ].
- 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 EKU field.
  - OCSP certificates presented for OCSP responses shall have the OCSP Signing purpose (id-kp 9 with OID 1.3.6.1.5.5.7.3.9) in the EKU 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 EKU field.

Application Note: Addressed TD0641.

FIA_X509_EXT.1.2(1) The TSF shall [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.

5.4.4.6 FIA_X509_EXT.2 X.509 Certificate Authentication *(MDM_PP)*

FIA_X509_EXT.2.1 The TSF shall [

- implement functionality to use X.509v3 certificates as defined by RFC 5280 to support authentication for [
  - TLS as defined in the Package for Transport Layer Security,

  ], and [
  - policy signing

  ].

]

Application Note: Addressed TD0784.

FIA_X509_EXT.2.2 When the [TSF] cannot establish a connection to determine the validity of a certificate, the TSF shall [implement functionality] to [not accept the certificate].

5.4.5 Security Management (FMT)

5.4.5.1 FMT_MOF.1(1) Management of Functions Behaviour *(MDM_PP)*

FMT_MOF.1.1(1) **Refinement:** 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].

5.4.5.2 FMT_MOF.1(2) Management of Functions Behaviour (Enrollment) *(MDM_PP)*

FMT_MOF.1.1(2) **Refinement:** The **MDM Server** shall restrict the ability to [initiate the enrollment process] to [authorized administrators and MD users].

5.4.5.3 FMT_MOF.1(3) Management of Functions in (MAS Server Downloads) *(MDM_PP)*

FMT_MOF.1.1(3) Refinement: 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.

5.4.5.4 FMT_POL_EXT.1 Trusted Policy Update *(MDM_PP)*

FMT_POL_EXT.1.1 The TSF shall provide digitally signed policies and policy updates to the MDM Agent.

FMT_POL_EXT.1.2 The TSF shall sign policies and policy updates using a private key associated with [an X.509 certificate] trusted by the agent for policy verification.

Application Note: Addressed TD0754.

FMT_POL_EXT.1.3 For each unique policy managed by the TSF, the TSF shall validate that the policy is appropriate for an agent using [client authentication via an X509 certificate representing the agent].

Application Note: Addressed TD0754.

5.4.5.5 FMT_POL_EXT.2 Agent Trusted Policy Update *(AGT_PPM)*

FMT_POL_EXT.2.1 The MDM Agent shall only accept policies and policy updates that are digitally signed by a private key 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 signature check fails.

Application Note: Addressed TD0755.

5.4.5.6 FMT_SMF.1(1) Specification of Management Functions (Server configuration of Agent) *(MDM_PP)*

FMT_SMF.1.1(1) **Refinement:** 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:

[

- no other management functions

] **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 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 additional 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 and microphone*] across device, [
    - no other method

    ], (MDF Function 5)

and the following MD configuration policies: [
60. [no other policies]
]
]

5.4.5.7 FMT_SMF.1(2) Specification of Management Functions (Server Configuration of Server) *(MDM_PP)*

FMT_SMF.1.1(2) **Refinement:** The TSF shall be capable of performing the following management functions:

b. configure the [
- devices specified by [*[a unique device ID]*],
- a number of devices,

] and [no other features] allowed for enrollment

c. [
9. No other management functions

].

Application Note: Modified per TD0887.

5.4.5.8 FMT_SMF.1(3) Specification of Management Functions (MAS Server) *(MDM_PP)*

FMT_SMF.1.1(3) **Refinement**: The MAS Server shall be capable of performing the following management functions:

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

5.4.5.9 FMT_SMF_EXT.4 Specification of Management Functions *(AGT_PPM)*

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],
- [no additional functions].

Application Note: Addressed TD0755.

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].

5.4.5.10 FMT_SMR.1(1) Security Management Roles *(MDM_PP)*

FMT_SMR.1.1(1) **Refinement:** The TSF shall maintain the roles

- administrator,
- MD user, and [
- *[the two MAS Server roles identified in FMT_SMR.1(2)]*

].

FMT_SMR.1.2(1) The TSF shall be able to associate users with roles.

5.4.5.11 FMT_SMR.1(2) Security Management Roles (MAS Server) *(MDM_PP)*

FMT_SMR.1.1(2) **Refinement:** The TSF shall additionally maintain the roles
- enrolled mobile devices,
- application access groups, and
- [*no additional authorized identified roles*].

FMT_SMR.1.2(2) **Refinement:** The MAS Server shall be able to associate users with roles.

5.4.5.12 FMT_UNR_EXT.1 User Unenrollment Prevention *(AGT_PPM)*

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

].

5.4.6 Protection of the TSF (FPT)

5.4.6.1 FPT_API_EXT.1 Use of Supported Services and APIs *(MDM_PP)*

FPT_API_EXT.1.1 The TSF shall use only documented platform API's.

5.4.6.2 FPT_ITT.1(2) Internal TOE TSF Data Transfer (MDM Agent) *(MDM_PP)*

FPT_ITT.1(2) Refinement: The TSF shall [
- implement functionality using [
  - mutually authenticated TLS as defined in the Package for Transport Layer Security,
  - HTTPS in accordance with FCS_HTTPS_EXT.1

  ]

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

Application Note: Because this TSF is distributed the "between the TSF and MDM Agent" above refers to "between the MDM Server and the MDM Agent".

5.4.6.3 FPT_LIB_EXT.1/AGT Use of Third Party Libraries *(MDM_PP)*

FPT_LIB_EXT.1.1/AGT **The MDM software shall be packaged with only [**
- *Ivanti MDM Android Client OpenSSL Component Cryptographic Library, v2.2.1,*
- *Refer to Appendix B for OTS Libraries*

].

5.4.6.4 FPT_LIB_EXT.1/SRV Use of Third Party Libraries *(MDM_PP)*

FPT_LIB_EXT.1.1/SRV **The MDM software shall be packaged with only [**
- *Apache 2.4,*
- *Ivanti MDM Bouncy Castle, v1.0.2.4,*
- *Ivanti MDM OpenSSL Component, 1.1.1g,*
- *Refer to Appendix B for OTS Libraries*

].

5.4.6.5 FPT_TST_EXT.1 Functionality Testing (MDM_PP)

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 [
- implement 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 [
- TSF,
- TOE platform

]-provided cryptographic services.

Application Note: TD0438 removed the requirement for Agents in a distributed TOE to perform self-tests.

5.4.6.6 FPT_TUD_EXT.1 Trusted Update *(MDM_PP)*

FPT_TUD_EXT.1.1 The TSF shall provide Authorized Administrators the ability to query the current version of the software.

FPT_TUD_EXT.1.2 The TSF shall [implement functionality] to provide Authorized Administrators the ability to initiate updates to TSF software.

FPT_TUD_EXT.1.3 The TSF shall [implement functionality] to provide a means to verify software updates to the TSF using a digital signature mechanism prior to installing those updates.

Application Note: Addressed TD0438.

5.4.7 Trusted Path/Channels (FTP)

5.4.7.1 FTP_ITC.1(1) Inter-TSF Trusted Channel (Authorized IT Entities) *(MDM_PP)*

FTP_ITC.1.1(1) **Refinement:** The TSF shall [
- implement functionality using [
  - mutually authenticated TLS as defined in the Package for Transport Layer Security,

]

] to provide a trusted communication channel between itself and authorized IT entities supporting the following capabilities:
- audit server,
- [*[no other capabilities]*]

that is logically distinct from other communication channels and provides assured identification of its end points and protection of the channel data from modification and disclosure.

FTP_ITC.1.2(1) **Refinement:** The TSF shall [implement functionality] to permit the MDM Server or other authorized IT entities to initiate communication via the trusted channel.

FTP_ITC.1.3(1) **Refinement:** The TSF shall [implement functionality] to initiate communication via the trusted channel for [

- *Sending MDM Server audit events to the Audit Server,*

].

Application Note: Addressed TD0784.

5.4.7.2 FTP_ITC_EXT.1 Trusted Channel *(MDM_PP)*

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)].

5.4.7.3 FTP_TRP.1(1) Trusted Path (for Remote Administration) *(MDM_PP)*

FTP_TRP.1.1(1) **Refinement:** The TSF shall **[**

- implement functionality using [
  - TLS as defined in the Package for Transport Layer Security,
  - HTTPS in accordance with FCS_HTTPS_EXT.1,

  ]

] 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) **Refinement:** The TSF shall [implement functionality] to permit remote administrators to initiate communication via the trusted path.

FTP_TRP.1.3(1) **Refinement:** The TSF shall [implement functionality] to require the use of the trusted path for [all remote administration actions].

Application Note: Applied TD0784.

5.4.7.4 FTP_TRP.1(2) Trusted Path (for Enrollment) *(MDM_PP)*

FTP_TRP.1.1(2) **Refinement: The TSF shall [**

- implement functionality using [
  - TLS as defined in the Package for Transport Layer Security,
  - HTTPS in accordance with FCS_HTTPS_EXT.1

  ]

] 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) **Refinement:** The TSF shall [implement functionality] to permit MD users to initiate communication via the trusted path.

FTP_TRP.1.3(2) **Refinement:** The TSF shall [implement functionality] to require the use of the trusted path for [all MD user actions].

5.5 TOE SFR Dependencies Rationale for SFRs

The PP and any relevant EPs/Modules/Packages contain(s) all the requirements claimed in this ST. As such, the dependencies are not applicable since the PP has been approved.

5.6 Security Assurance Requirements (SARs)

The TOE assurance requirements for this ST are taken directly from the PP and any relevant EPs/Modules/Packages, which are derived from Common Criteria Version 3.1, Revision 5. The assurance requirements are summarized in Table 19.

Table 19: Security Assurance Requirements

Assurance Class	Assurance Components	Component Description
Security Target	ASE_CCL.1	Conformance claims
	ASE_ECD.1	Extended components definition
	ASE_INT.1	ST introduction
	ASE_OBJ.1	Security objectives for the operational environment
	ASE_REQ.1	Stated security requirements
	ASE_SPD.1[2]	Security problem definition
	ASE_TSS.1	TOE Summary Specification
Development	ADV_FSP.1	Basic functionality specification
Guidance Documents	AGD_OPE.1	Operational user guidance
	AGD_PRE.1	Preparative Procedures
Life Cycle Support	ALC_CMC.1	Labelling of the TOE
	ALC_CMS.1	TOE CM coverage
Tests	ATE_IND.1	Independent testing – conformance
Vulnerability Assessment	AVA_VAN.1	Vulnerability survey

5.7 Assurance Measures

The TOE satisfied the identified assurance requirements. This section identifies the Assurance Measures applied by Ivanti, Inc. to satisfy the assurance requirements. The following table lists the details.

[2] ASE_SPD.1 is missing from Table 3 of the PP_MDM_V4.0. However, it is included due to the statement in 7.1 Class ASE: Security Target "The ST is evaluated as per ASE activities defined in the CEM".

Table 20: TOE Security Assurance Measures

SAR Component	How the SAR will be met
ADV_FSP.1	The functional specification describes the external interfaces of the TOE; such as the means for a user to invoke a service and the corresponding response of those services. The description includes the interface(s) that enforces a security functional requirement, the interface(s) that supports the enforcement of a security functional requirement, and the interface(s) that does not enforce any security functional requirements. The interfaces are described in terms of their purpose (general goal of the interface), method of use (how the interface is to be used), parameters (explicit inputs to and outputs from an interface that control the behavior of that interface), parameter descriptions (tells what the parameter is in some meaningful way), and error messages (identifies the condition that generated it, what the message is, and the meaning of any error codes).
AGD_OPE.1	The Administrative Guide provides the descriptions of the processes and procedures of how the administrative users of the TOE can securely administer the TOE using the interfaces that provide the features and functions detailed in the guidance.
AGD_PRE.1	The Installation Guide describes the installation, generation, and startup procedures so that the users of the TOE can put the components of the TOE in the evaluated configuration.
ALC_CMC.1 ALC_CMS.1	The Configuration Management (CM) documents describe how the consumer identifies the evaluated TOE. The CM documents identify the configuration items, how those configuration items are uniquely identified, and the adequacy of the procedures that are used to control and track changes that are made to the TOE. This includes details on what changes are tracked and how potential changes are incorporated.
ATE_IND.1	Vendor will provide the TOE for testing.
AVA_VAN.1	Vendor will provide the TOE for testing. Vendor will provide a document identifying the list of software and hardware components.

6 TOE Summary Specification

6.1 Security Audit (FAU)

6.1.1 FAU_ALT_EXT.1 Server Alerts *(MDM_PP)*

The MDM Server component of the TOE supports sending email Alerts. These alerts can also be viewed by an administrator on the MDM server. The MDM Server will alert the admin of the following changes:

- Changes to enrolled devices.
  - Device Registered Alert: This alert is sent when a device is registered. Specifically, the Mobile Device user will be provided with the MDM Server's FQDN and login credentials instructing the user on how to enroll in the EPMM system.
  - Unenroll Alert: This alert is sent when a device is unenrolled (retired). Unenrollment is accomplished by the MDM Server sending a Wipe command to the device. Once wipe is initiated the device factory resets, thereby making the device to the preconfigured mode i.e. unenrolled.
- Failure to apply policies to a mobile device
  - Policy Update Failure Alert: This alert is sent to the administrator when a policy change issued by the MDM server fails to be applied on the MDM Agent. This alert can be viewed by the administrator under the devices tab where it will indicate that the policy is either not applied or unsupported.

The Alert will be sent to the email address of the Android MD User configured from the System Manager Add New User option and assigned to a device when the device is created from the Admin Portal Add Single Device option. An alert is different from an audit record, however the fact that an alert was sent is audited per FAU_GEN.1. The MDM Server includes the Type of Alert (ACTION) generated and the ID of the Mobile Device involved in the audit record.

6.1.2 FAU_ALT_EXT.2 Agent Alerts *(AGT_PMM)*

The TOE supports the ability to periodically synchronize the MDM Server and MDM Agents. The synchronization includes retrieving information about the policies that are installed, thereby ensuring the MDM Server is informed about which policies have been applied.

The Agents, in the evaluated configuration, are configured to check in every eight hours but this parameter can be modified.

Refer to FMT_POL_EXT.1 below for a description of how policy updates are obtained and the actions that take place for success and unsuccessful cases.

When a mobile device checks in, the mobile device performs a compliance check based on configured policies. If any of the settings have not been reported to and acknowledged by the MDM Server, the mobile device reports those changes. Hence, if something happens, such as a network disruption, that prevents the MDM Server from receiving the mobile device compliance information or prevents the mobile device from receiving an acknowledgement from the MDM Server, that information will be sent the next time the device connects until it is finally acknowledged.

The MDM Agent will store unsent alerts in the application storage available on the mobile device and that space is limited only by the space available on the mobile device.

6.1.3 FAU_CRP_EXT.1 Support for Compliance Reporting of Mobile Device Configuration *(MDM_PP)*

The EPMM TOE supports the ability of the MDM Server to receive the following information from the MDM Agents:

- The current version of the MD Ivanti Agent software.
- The current version of the hardware model of the device.
- The current version of installed mobile applications.
- List of MD configuration policies that are in place on the device.

This information is transmitted across a dedicated secure channel implementing mutually authenticated HTTPS (HTTP over TLS) with the MDM Server functioning as the TLS Server and the mobile device (MDM Agent) functioning as the TLS Client.

If the mobile device is eligible, the MDM Server will send a Push Request to the Notification Server which in turn sends the Push Request to the MDM Agent.

6.1.4 FAU_GEN.1(1) Audit Data Generation *(MDM_PP)*

The MDM Server component of the TOE produces audit events reporting multiple activities. The following types of audit events are supported by the MDM Server:

- **Administrative Actions**: the TSF will produce an audit record for every Administrative command issued by an Administrator from the Admin Portal and the System Manager.
- **Required Audit Events**: This type includes every required audit record identified by the PP and Module. Specifically,
    - The defined auditable events listed in **Table 16** required by the MDM Server**.**
    - The defined auditable events listed in FAU_GEN.1(2)/SRV required by the MAS functionality.

Refer to FAU_GEN.1(2)/AGT for a list of the audit events generated by the MDM Agent.

Appendix A identifies the list of every audit event type mandated by the PP_MDM_V4.0 and MOD_MDM_AGENT_V1.0.

Each audit entry not only identifies the security incident but also includes

- a timestamp,
- type of event,
- user identity when applicable,
- the success or failure of the event, and
- who generated the audit event (TSF or TOE platform).

Audit records generated by the MDM Server that are displayed via the Admin Portal Logs GUI page have the following format:

ACTION | STATE | PERFORMD BY | ACTION DATE | COMPLETED AT | PERFORMED ON

The information panel displays:

- ACTION - a short description of the event.
- STATE – success or failure.

- PERFORMED BY – login of person who caused the event.
- ACTION DATE – date the event was started. Specifically, day of week | month | day | year | time | time zone.
- COMPLETED AT – date the event occurred. Specifically, day of week | month | day | year | time | time zone.
- PERFORMED ON – the system the event occurred on.
- DETAILS – displays the outcome of the event.

Audit records generated by the MDM Server that are sent to the syslog server have the following format. The following is an example of a syslog message and the fields are described below.

<13>Oct 22 12:34:56 myhostname myapp[1234] This is a sample syslog message.

- <13> = **Priority**: A numerical value that combines the facility and severity level of the message. It is enclosed in angle brackets.
- Oct 22 12:34:56 = **Timestamp**: Indicates the time when the message was generated. It is typically in the format of MMM DD HH:MM:SS (month, day, hour, minute, second).
- myhostname = **Hostname:** Identifies the origin of the syslog message, indicating the system or device that generated the log.
- myapp[1234] = **Tag:** Tag or Process ID represents the application or process that generated the message. It is often user-defined.
- This is a sample syslog message = **Message:** The actual log information or event description.

Table 16: MDM Server Security Functional Requirements and Auditable Events Table 16 includes a list of the audit records generated by the MDM Server minus the audit records generated for MAS functionality.

The TOE is a distributed TOE and therefore, if more than one TOE component is involved when an audit event is generated, the event is audited on each component.

6.1.5 FAU_GEN.1(2)/AGT Audit Data Generation *(AGT_PPM)*

The MDM Agent component of the TOE provides the following audit records for

- Startup and shutdown of the MDM Agent.
- MDM Agent is enrolled (FIA_ENR_EXT.2).
- A policy update,
- Failure of the randomization process. (FCS_RBG_EXT.1/AGT).

Any modification commanded by the MDM Server specifically all events listed in Table 17: EPMM Agent *(AGT_PPM)* Security Functional Requirements and Auditable Events.

Audit records generated by the MDM Agent that are generated have the following format. The following is an example of an audit message and the fields are described below.

2025-05-21 16:26:47.99 32754 676 E=OpenSSLWrapper: error:0A0000F8:SSL routines:::unknown cipher returned in ssl/statem/statem_clnt c (set_client_ciphrsuite): 1313 data: flages:

- 2025-05-21 = **Date**: the date the message was generated (year-month-day).
- 16:26:47.99 = **Timestamp**: Indicates the time when the message was generated. It is in the format of HH:MM:SS (hour, minute, second).
- 32754 = **Process ID:** Process ID represents the application or process that generated the message.

- 676 = **Message ID**: A numerical value that represents the message ID.
- E= **Severity**: The Severity value of the message. Available values are:
  - **A**lert
  - **C**ritical
  - **E**rror (the default)
  - **W**arning
  - **N**otice
  - **D**ebug
  - **I**nformation
- OpenSSLWrapper:…. = **Message:** The actual log information or event description.

The information displayed as Process ID and Message ID will vary based on the Severity of the audit record.

Agent audit records are identified in Appendix A.

6.1.6 FAU_GEN.1(2)/SRV Audit Generation (MAS Server) *(MDM_PP)*

The MDM Server component of the TOE supports MAS Server functionality. The Server will generate an audit record for the following MAS Server events:

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

The MAS Server functionality of the TOE will include the following information in each MAS Server event:

- date and time of the event,
- type of event, and
- mobile device identity.

MAS Server events generated by the Admin Portal are available for view using the Admin Portal > Logs option. Refer to 6.1.4 FAU_GEN.1(1) Audit Data Generation *(MDM_PP)* for a description of the format of the audit logs.

6.1.7 FAU_NET_EXT.1 Network Reachability Review *(MDM_PP)*

Admin Portal Administrators can view whether an Android device is connected or not by viewing the Device & Users > Device page. Each enrolled Android device will display whether the device is Active or Not Active in the STATUS column available from the Admin Portal Devices & Users > Devices GUI.

The status is always present and can be viewed from the GUI screen when an administrator logs onto the MDM Server because it is information exchanged during a device check-in. The information is as current as the value of the Sync Interval parameter in the device's Sync Policy (received via an MDM Agent initiated event). An administrator may select to view current information by selecting the Action > Force Device Check-In for a specific device from the Admin Portal. This will force a device check-in and update the above parameters as well as all new policies (received via an MDM Server initiated event).

6.1.8 FAU_SAR.1 Audit Review *(MDM_PP)*

The MDM Server component of the TOE provides two GUI administrator interfaces that enables an administrator to perform administrative actions that include viewing audit logs generated by the MDM Server. Specifically, audit records generated by the Admin Portal functions are available for viewing via the Admin Portal > Logs option and audit records generated by the System Manager functions are available for viewing via the System Manager > Logs option.

Audit records generated by the MDM Server (FAU_GEN.1(1) and FAU_GEN.1(2)/SRV) are viewed by the Server Administrator by selecting the Logs option from the Admin Portal.

Audit records generated by the MDM Agent (FAU_GEN.1(2) are viewed by the Server Administrator by pulling the device logs from the Agent. Once resident on the MDM Server, the logs can be viewed by the Server Administrator.

6.1.9 FAU_SEL.1(2) Security Audit Event Selection *(AGT_PPM)*

The MDM Agent's Android 13 and Android 14 operating system (OS) (Operational Environment) and the MDM Agent application (TOE) both generate audit records.

The MDM Agent provides both the platform provided system logs and implements system logging (application logs of agent) and, therefore, includes events and logs from both. The MDM Agent component of the TOE stores most audit records in the mobile device audit log and thereby leverages the selection functions of the mobile device to allow audited events to be selected based on the following attributes: event type; success of auditable security events; failure of auditable security events to the operational environment.

The following events are configurable available through the Samsung General Policy Audit Events option.

i. CA certificate import
ii. CA certificate removal
iii. Request current software version
iv. Request current hardware version
v. Report app inventory
vi. Upload logs
vii. Enable/Disable camera
viii. Enable/Disable microphone

The other auditable events included in the list that are considered security relevant and therefore not subject to the selection criteria are:

i. Audit configuration change
ii. TLS session failure
iii. Transition to locked state
iv. Enrollment in management
v. Unenrollment from management
vi. Policy validation failure
vii. ID certificate import
viii. ID certificate removal
ix. Install application
x. Remove application
xi. Device OS upgrade
xii. Server request client check-in
xiii. Configure App Store quarantine compliance action

The remaining events (e.g., policy signature related audits and agent alerts) not stored in the platform audit log are always audited (i.e., cannot be de-selected) and are stored locally by the MDM Agent in its data space until delivered to the MDM Server.

The Administrator is required to configure the level of auditing required for the evaluated configuration. The Samsung General Policy Severity Rule parameter must be set to Error (the default). This setting will ensure that all events with the severity level of Error, Critical, and Alert will be displayed.

6.1.10 FAU_STG_EXT.1 External Trail Storage *(MDM_PP) and* FAU_STG_EXT.2 Audit Event Storage *(MDM_PP)*

For distributed TOEs, each component must be able to export audit data across a protected external channel or an intercomponent channel.

The MDM Server component of the TOE stores audit records locally and supports the functionality to transmit the audit records via an external TOE trusted channel to an Audit Server running Syslog. At TOE installation, the evaluated configuration requires a Syslog Server to be configured on a remote host. Upon startup, the TOE connects to the Syslog Server using the Bouncy Castle cryptographic library (Line E Figure 1). If the Syslog Server's certificate is successfully validated, the TOE starts another connection to the Syslog Server using the Ivanti MDM OpenSSL crypto library (Line F Figure 1). This is the connection used to continuously transfer audit records to the Syslog Server. If this connection goes down, it is retried (the Bouncy Castle connection is not). The Bouncy Castle Syslog Server is only retried at next system reboot.

The MDM Agent component of the TOE stores audit records locally. The MDM Server's Admin Portal provides functionality that enables an Administrator to pull audit data from the Android devices. The function is an Action, which means it happens immediately vs at check-in. The command is Actions > Pull Client Logs from the Admin Portal. The TSF uses the mutually authenticated trusted channel (FPT_ITT.1(2)) to transfer the audit data at the request of the MDM Server Administrator (Line B Figure 1). The MDM Agent users (MD Users) are managed by the component's operating system (operational environment) and therefore, any actions they can perform on audit records is out scope for the evaluation.

Once the agent's audit records are local to the MDM Server, the TOE uses the System Manager to download the agent's audit records to the Audit Server. For this connection, the TOE, at installation, is configured to connect via HTTPS to the Audit Server. To transfer the records, the System Manager Administrator navigates to the Troubleshooting tab and selects Export Logs > Download > Export option. This selection will transfer the MDM Agent's log files to the Audit Server via HTTPS. The communication channel used to export the files is the FTP_ITC.1(1) Inter-TSF Trusted Channel (Authorized IT Entities) *(MDM_PP)* (Line G Figure 1).

In summary, the MDM Server supports three TLS Client channels (Line E Figure 1) (Bouncy Castle to Syslog), (Line F Figure 1) (OpenSSL to Syslog) and (Line G Figure 1) (OpenSSL to HTTPS) Figure 1) to transfer audit data to the Audit Server. The Audit Server supports both HTTPS and Syslog to support the export of audit data.

The MDM Server component of the TOE provides the functions necessary for an administrator to review all of the collected audit records, while ensuring that the audit records cannot be modified. The MDM Server doesn't offer any functions that allow the audit log or individual audit records therein to be modified or inserted. The MDM server does offer authorized Administrators a way to clear the audit data. This functionality is used to delete MDM Agent audit records, resident on the MDM Server, after they were exported to the Audit Server.

6.2 Communication (FCO)

6.2.1 FCO_CPC_EXT.1 Component Registration Channel Definition *(MDM_PP)*

For a description of MDM Agent registration channels are implemented refer to Section 6.4.2 FIA_ENR_EXT.1 Enrollment of Mobile Device into Management *(MDM_PP).*

For a description of audit record protection functionality is invoked refer to Section 6.7.1 FTP_ITC.1(1) Inter-TSF Trusted Channel (Authorized IT Entities) *(MDM_PP).* This includes transferring MDM Agent audit records from MDM Agents to MDM Servers on the Internal TOE TSF Data Transfer trusted channel (FPT_ITT.1(2)).

Note: The TSS Section of the PP is incorrectly requesting a description of how the audit record protection is invoked. The SFR calls out the FPT_ITT.1(2) which is the data transfer trusted channel between the TOE components.

6.3 Cryptographic Support (FCS)

6.3.1 FCS_CKM.1 Cryptographic Key Generation *(MDM_PP)*

The TOE supports RSA key generation schemes as specified in FIPS PUB 186-5, with key sizes of 2048 bits, 3072 bits, and 4096. The TOE also supports ECC schemes using “NIST curves” P-256, P-384, and P-521 as specified in FIPS PUB 186-5, Appendix B.4.

The keys are generated for key establishments used in establishing a TLS communication channel. The MDM Server acts as both the sender and the receiver in the key establishment. The MDM Agent acts as the sender of key establishment.

Both the MDM Server and MDM Agent components of the TOE implement the key generation algorithms by including the cryptographic libraries used to generate keys in the TOE’s physical boundary. The MDM Servers use the Ivanti MDM OpenSSL Component cryptography library for all TLS paths/channels except the initial TLS connection to the Syslog Server. For that connection, Ivanti MDM Bouncy Castle cryptographic library is used. The MDM Agents use the Ivanti MDM Android Client OpenSSL Component cryptography library. Both Android platforms perform identically. All three libraries are included in the TOE physical boundary and CAVP certified as detailed in Table 28: CAVP Algorithm Certificate References.

6.3.2 FCS_CKM.2 Cryptographic Key Establishment *(MDM_PP)*

The TOE supports the following key establishment schemes, including:

- RSA based that meets 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",
- ECC based key exchange based on NIST Special Publication 800-56A Revision 2, “Recommendation for Pair-Wise Key Establishment Schemes Using Discrete Logarithm Cryptography”.

The key generation schemes used are RSA and ECC (elliptic curve) based and the key establishment schemes coincides with selections made in FCS_CKM.1, described above. The key establishment is used in establishing a TLS communication channel.

The MDM Server component of the TOE acts as a sender and receiver for both schemes. The MDM Agent component of the TOE acts as a sender for both schemes.

The TOE handles decryption errors in accordance with NIST Special Publication 800-56B. The TOE does not reveal the particular error that occurred, either through the contents of any output, any logged error

message, or through timing variations. In the event that a decryption error occurs, no connection will be established. This would be detected and failed at the TLS handshake. No error is logged and no message is output in the case of a failure.

Both the MDM Server and MDM Agent components of the TOE implement the key establishments algorithms by including the cryptographic libraries used to establish keys in the TOE's physical boundary. The MDM Servers use the Ivanti MDM OpenSSL Component cryptography library for all TLS paths/channels except the initial TLS connection to the Syslog Server. For that connection, Ivanti MDM Bouncy Castle cryptographic library is used. The MDM Agents use the Ivanti MDM Android Client OpenSSL Component crypto library defined in Table 28: CAVP Algorithm Certificate References

6.3.3 FCS_CKM_EXT.4 Cryptographic Key Destruction (MDM_PP)

The MDM Agent invokes platform provided functionality for plaintext key destruction in volatile and non-volatile memory by overwriting with zeroes.

The MDM Server implements plaintext key destruction in volatile and non-volatile memory through zeroization method depending on the type of the memory. All persistent keys stored in the non-volatile memory are encrypted using AES GCM.

The TOE meets all requirements specified in FIPS 140-3 for destruction of keys and Critical Security Parameters (CSPs). All keys within the TOE are securely destroyed as per the descriptions given in the following table.

Table 21: Keys and CSPs

Component	Key or CSP	Storage Location	Stored form	Destruction
MDM Agent	Agent public and private key	Non-volatile storage (Platform key storage)	Plaintext	Logically addresses the storage location of the key and overwriting with zeroes by invoking underlying platform provided interface.
	Agent public and private key	Volatile memory (in use)	Plaintext	Overwritten with zeroes when the TLS session between the Agent and MDM server terminates.
	TLS session key	Volatile memory	Plaintext	Overwritten with zeroes when the TLS session terminates.
	Mobile device user credential	Non-volatile storage (Platform key storage)	Plaintext	Logically addresses the storage location of the key and overwriting with zeroes by invoking underlying platform provided interface.
MDM Server	Administrator / User credential	Non-volatile storage	Encrypted	Persistent storage is in encrypted form using AES GCM.
	Web Portal public and private key	Non-volatile storage	Encrypted	Persistent storage is in encrypted form using AES GCM.
	Web Portal public and private key	Volatile memory (in use)	Plaintext	Overwritten with zeroes when a new certificate is loaded.

	CA public and private keys	Non-volatile storage	Encrypted	Persistent storage is in encrypted form using AES GCM.
	CA public and private keys	Volatile memory (in use)	Plaintext	Overwritten with zeroes when a new certificate is loaded.
	TLS session key	Volatile memory	Plaintext	Overwritten with zeroes when the TLS session terminates.
	Initialization Vector for AES GCM	Volatile memory	Plaintext	Overwritten with zeroes.

TLS private and public keys are destroyed when the associated X509v3 certificate is removed from the trust store of the TOE. All other keys are ephemeral, and are destroyed by the TOE when their associated session is terminated.

The TOE does not make use of a value that does not contain any CSP to overwrite keys. The TOE does not have any circumstances that may not conform to key destruction requirements.

Both the MDM Server and MDM Agent components of the TOE implement the functionality to manage keys used for TLS session by including the cryptographic libraries used to manage the keys in the TOE's physical boundary. Key destruction does not require the algorithms to be CAVP certified, however the cryptographic libraries do include the functionality. The MDM Servers use the Ivanti MDM OpenSSL Component cryptography library for all TLS paths/channels except the initial TLS connection to the Syslog Server. For that connection, Ivanti MDM Bouncy Castle cryptographic library is used. The MDM Agents use the Ivanti MDM Android Client OpenSSL Component cryptography library. Both Android platforms perform identically. All three libraries are included in the TOE physical boundary.

6.3.4 FCS_COP.1.1(1) Cryptographic Operation (Confidentiality Algorithms) *(MDM_PP)*

The TOE provides cryptographic encryption and decryption services as specified in NIST SP 800-38D and FIPS Pub 180-4. Encryption and decryption is used for the following:

- TLS communication to ensure data confidentiality, integrity, and authentication.
- To store X.509 certificates used by the MDM Server.

The TSF performs cryptographic encryption and decryption with the following algorithms and message digest sizes:

- AES-GCM
- with cryptographic key size of 128-bit and 256-bit.

In the event of decryption errors, particularly for communication (e.g., key establishment), the associated function for both the server and agent will fail and be logged, where appropriate, as a higher level session failure with no specific details about the decryption failure being disclosed.

Both the MDM Server and MDM Agent components of the TOE implement the encryption and decryption algorithms by including the cryptographic libraries used to encryption and decryption in the TOE's physical boundary. The MDM Servers use the Ivanti MDM OpenSSL Component cryptography library for all TLS paths/channels except the initial TLS connection to the Syslog Server. For that connection, Ivanti MDM Bouncy Castle cryptographic library is used. Additionally, Ivanti MDM Bouncy Castle cryptographic library is used by the MDM Servers to encrypt data-at-rest (keys and certificates) stored by the MDM Server. The

MDM Agents use the Operational Environment's Android Keystore system to manage and store keys and certificates. This system implements the Android's cryptographic library (Operational Environment) to implement encryption and decryption, Samsung BoringSSL Android 1.7 and Samsung BoringSSL Android 1.8 for Android 13 and Android 14 respectively. The MDM Agents use the Ivanti MDM Android Client OpenSSL Component cryptography library. All three libraries are included in the TOE physical boundary and CAVP certified as detailed in Table 28: CAVP Algorithm Certificate References.

6.3.5 FCS_COP.1(2) Cryptographic Operation (Hashing Algorithms) *(MDM_PP)*

The TOE provides cryptographic hashing services as specified in FIPS Pub 180-5 Hashing is used for the following:

- TLS communication,
- Digital signing (MDM Server) and verification (MDM Agent) of Policies,
- Digital signature verification by both TOE components as part of trusted update validation.

The TSF performs cryptographic hashing with the following algorithms and message digest sizes:

- SHA-256
- SHA-384
- SHA-512
- and message digest sizes of 256 bits, 384 bits, and 512 bits respectively.

The TOE uses a byte-oriented mode for hashing.

Both the MDM Server and MDM Agent components of the TOE implement the hash functionality by including the cryptographic libraries used to generate hashes in the TOE's physical boundary. The MDM Servers use the Ivanti MDM OpenSSL Component cryptography library for all TLS paths/channels except the initial TLS connection to the Syslog Server. For that connection, Ivanti MDM Bouncy Castle Cryptographic Library is used. The MDM Agents use the Ivanti MDM Android Client OpenSSL Component cryptography library. All three libraries are included in the TOE physical boundary and CAVP certified as detailed in Table 28: CAVP Algorithm Certificate References.

6.3.6 FCS_COP.1(3) Cryptographic Operation (Signature Algorithms) *(MDM_PP)*

The TSF uses the RSA scheme using key sizes of 2048, 3072, and 4096 bits and ECDSA using P-256, P-384 and P-521 curves. RSA schemes meet FIPS PUB 186-5, 'Digital Signature Standard (DSS)', Section 4. ECDSA Schemes meet FIPS PUB 186-5, 'Digital Signature Standard (DSS)', Section 5.

The TOE provides MDM Servers and MDM Agent the ability to verify software updates to the TSF.

Additionally, Policies received by the MDM Agents are digitally signed by a certificate that has been authorized for policy updates by the MDM Server. This is the certificate that is sent by the MDM Server to the MDM Agent at enrollment time. This is to ensure the received policies are tied to MDM Server and not to protect the policies in transit as they are protected by FPT_ITT.1(2).

Both the MDM Server and MDM Agent components of the TOE implement the signature generation and verification algorithms by including the cryptographic libraries used to generate and verify signatures in the TOE's physical boundary. The MDM Servers use the Ivanti MDM OpenSSL Component cryptography library for all TLS paths/channels except the initial TLS connection to the Syslog Server. For that connection, Ivanti MDM Bouncy Castle cryptographic library is used. The MDM Agents use the Ivanti MDM Android Client OpenSSL Component cryptography library. All three libraries are included in the TOE physical boundary and CAVP certified as detailed in Table 28: CAVP Algorithm Certificate References.

6.3.7 FCS_COP.1(4) Cryptographic Operation (Keyed-Hash Message Authentication) *(MDM_PP)*

The TOE provides keyed-hashing message authentication services as specified in FIPS Pub 198-1, "The Keyed-Hash Message Authentication Code," and FIPS 180-4, "Secure Hash Standard."

The TSF performs keyed-hash message authentication using the following:

Table 22: HMAC Details

Function	Hash Algorithm	Block Size	Key Length	Message Digest Size
HMAC-SHA-256	SHA-256	512 bits	256 bits	256 bits
HMAC-SHA-384	SHA-384	1024 bits	384 bits	384 bits

Both the MDM Server and MDM Agent components of the TOE implement the keyed hash algorithms by including the cryptographic libraries used to perform keyed hashing in the TOE's physical boundary. The MDM Servers use the Ivanti MDM OpenSSL Component cryptography library for all TLS paths/channels except the initial TLS connection to the Syslog Server. For that connection, Ivanti MDM Bouncy Castle cryptographic library is used. The MDM Agents use the Ivanti MDM Android Client OpenSSL Component cryptography library. All three libraries are included in the TOE physical boundary and CAVP certified as detailed in Table 28: CAVP Algorithm Certificate References.

6.3.8 FCS_HTTPS_EXT.1 HTTPS Protocol`*(MDM_PP)*

Both the MDM Server and MDM Agent components of the TOE support the HTTPS protocol (compliant with RFC 2818). Refer to Table 23: EPMM TLS Protocol Implementation for the itemized list of TLSC support.

MDM Server supports HTTPS Client communicating with the Audit Server and HTTPS Server when communicating with the MDM Agents and remote workstations.

6.3.9 FCS_IV_EXT.1 Initialization Vector Generation *(MDM_PP)*

Both the MDM Server and MDM Agent components of the TOE's crypto libraries include routines to generate an initialization vector (IV) used in encryption and decryption algorithms. All three libraries provide functions that enable the calling program to specify non-repeating or repeating, the length, and cipher mode. The TOE, using the library's RBG, generates AES-GCM IVs, that are non-repeating and 96-bits in length. The TOE uses IVs for the encryption/decryption algorithms used in TLS communication.

Administrator and user credentials, certificates, and private keys are stored persistently in flat files with specific permission to limit access to only the necessary components or processes. These flat files are encrypted with AES-GCM as defined in NIST SP 800-38D. The TOE uses IV for the AES GCM algorithm.

Refer to Section 6.3.12 FCS_STG_EXT.1 Cryptographic Key Storage *(MDM_PP)* and Section 6.3.13 FCS_STG_EXT.1(2) Cryptographic Key Storage *(AGT_PPM)* for the details of encryption of user credentials, persistent secrets, and private keys.

6.3.10 FCS_RBG_EXT.1/AGT Extended: Random Bit Generation *(MDM_PP)*

The MDM Agent provides an AES-256 CTR-DRBG (CAVP Cert #A6402) provided by the Ivanti MDM Android Client OpenSSL Component cryptographic library included in the TOE boundary.

6.3.11 FCS_RBG_EXT.1/SRV Extended: Random Bit Generation *(MDM_PP)*

The MDM Server obtains entropy from two sources. It utilizes Intel RDRAND which is a hardware based entropy noise source and the Linux Kernel Random Number Generator (LKRNG) which is a platform-based RBG for random bit generation services. These are in accordance with NIST special Publications 800-90A.

The MDM Server component of the TOE includes two cryptographic libraries: Ivanti MDM Bouncy Castle and Ivanti MDM OpenSSL Component. The OpenSSL library provides an AES-256 CTR_DRBG (CAVP #A6074) deterministic random bit generation. The Bouncy Castle library provides a SHA-256 Hash-DRBG (CAVP Cert #A6073).

Both libraries are included in the MDM Server software build. Both RBG's provide 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.

6.3.12 FCS_STG_EXT.1 Cryptographic Key Storage *(MDM_PP) and* FCS_STG_EXT.2 Encrypted Cryptographic Key Storage *(MDM_PP)*

The MDM Server component of the TOE uses it's Bouncy Castle cryptographic library along with the storage provided by the platform to keep persistent and private keys safe. The MDM Server stores its keys and certificates in files with specific permissions to limit access to only the necessary components or processes. Additionally, the files are encrypted with AES-GCM as defined in NIST SP 800-38D. Table 26 can be referenced for a list of all keys resident on the MDM Server, what they are used for and how they are stored.

Note: This SFR applies only to the MDM Server. The following SFR, FCS_STG_EXT.1(2) Cryptographic Key Storage *(AGT_PPM)* is mandatory and covers key storage for the MDM Agents.

6.3.13 FCS_STG_EXT.1(2) Cryptographic Key Storage *(AGT_PPM)*

The MDM Agent stores mobile device user credentials, private keys, and certificates persistently in the platform provided key storage (Android Key Store). This certificate is used to identify and authenticate the mobile device to the MDM Server on the TOE's internal TOE trusted channel (Line B Figure 1).

The platform provided Android Key Store encrypts the keys and certificates using the platform provided Operational Environment cryptographic library, which implements Samsung Boring SSL Android 1.7 or 1.8 for Android 13 and Android 14 respectively. Refer to Section 6.6.1 FTP_API_EXT.1 for a description of the API used to invoke the Android Key Store.

6.3.14 FCS_TLS_EXT.1 TLS Protocol *(TLS_PKG)*

The TOE provides both TLS Server and TLS Client functionality. The following table describe the TLS interfaces.

Table 23: EPMM TLS Protocol Implementation

TOE Component	Component's Implementation	Remote System	Description	Crypto Library	Figure 1 Identifier
MDM Server	HTTP over TLS Server	MDM Agent (TOE)	Trusted Path for Enrollment. FTP_TRP.1(2)	Ivanti MDM OpenSSL Component	Line A
	HTTP over TLS Server	MDM Agent (TOE)	Internal TOE TSF Data Transfer (MDM Agent). FPT_ITT.1(2)	Ivanti MDM OpenSSL Component	Line B
	HTTP over TLS Server	MDM Server Remote Workstation (OE)	Trusted Path for remote administration to the Admin Portal administrator management GUI. FTP_TRP.1(1)	Ivanti MDM OpenSSL Component	Line C
	HTTP over TLS Server	MDM Server Remote Workstation (OE)	Trusted Path for remote administration to the System Manager administrator management GUI. FTP_TRP.1(1)	Ivanti MDM OpenSSL Component	Line D
	TLS Client	Audit Server (OE)	Trusted Channel with Authorized IT Entities. . Initial Syslog Server connection made using Bouncy Castle Crypto Library. FTP_ITC.1(1) Protocol is TLS Mutual Authentication. Used for the first connection to the Syslog Server.	Ivanti MDM Bouncy Castle	Line E
	TLS Client	Audit Server (OE)	Trusted Channel with Authorized IT Entities. Second Syslog Server connection made using MDM OpenSSL Component Crypto Library. FTP_ITC.1(1) Used for the second connection to the Syslog Server.	Ivanti MDM OpenSSL Component	Line F
	HTTP over TLS Client	Audit Server (OE)	Trusted Channel with Authorized IT Entities. Protocol is HTTP/TLS.	Ivanti MDM	Line G

TOE Component	Component's Implementation	Remote System	Description	Crypto Library	Figure 1 Identifier
			FTP_ITC.1(1) Used to transfer MDM Agent audit records pulled from the MDM Agents and transferred to the Audit Server.	OpenSSL Component	
MDM Agent	HTTP over TLS Client	MDM Server (TOE)	Trusted Path for Enrollment. FTP_TRP.1(2)	Ivanti MDM Android Client OpenSSL Component	Line A
	HTTP over TLS Client	MDM Server (TOE)	Internal TOE Trusted Channel. FTP_ITT.1(2)	Ivanti MDM Android Client OpenSSL Component	Line B

6.3.15 FCS_TLSC_EXT.1 TLS Client Protocol *(TLS_PKG)*

The Ivanti EPMM TOE supports the TLS protocol (TLS 1.2 RFC 5246) for the purpose of protecting data in transit. Both the MDM Server component of the TOE and the MDM Agent component support TLSC functionality. Refer to Table 23: EPMM TLS Protocol Implementation for the itemized list of TLSC support.

When used for intra-TOE communication, TOE components support the following ciphersuites:

- TLS_RSA_WITH_AES_128_GCM_SHA256 as defined in RFC 5288,
- TLS_RSA_WITH_AES_256_GCM_SHA384 as defined in RFC 5288,
- TLS_ECDHE_ECDSA_WITH_AES_128_GCM_SHA256 as defined in RFC 5289,
- TLS_ECDHE_ECDSA_WITH_AES_256_GCM_SHA384 as defined in RFC 5289,
- TLS_ECDHE_RSA_WITH_AES_128_GCM_SHA256 as defined in RFC 5289,
- TLS_ECDHE_RSA_WITH_AES_256_GCM_SHA384 as defined in RFC 5289

If the TOE component receives a handshake message proposing an outdated version of TLS, the TOE will reject the connection. The TOE component will always propose only TLSv1.2.

The reference identifiers for TOE components or trusted IT entities are configured by the administrator using the available administrative commands in the Admin Portal. The reference identifier must be a DNS Name as described in RFC 6125 Section 6. This is true for both inter-TOE communications between TOE components and for communication with non-TOE trusted IT entities. The TOE doesn't support IP addresses as reference identifiers.

Note that when a mobile device is enrolling (Enrollment TLS Trusted Path Line A Figure 1) it is assigned a unique UUID that is associated with the serial number of the device's certificate. When the mobile device connects and presents its certificate (MA Inter-TSF Trusted Channel Line B Figure 1), the UUID in the certificate is used to look up the certificate serial number to ensure it matches the certificate that was presented.

When the TOE client receives an X.509 certificate from their respective servers, the client will compare the reference identifier with the established Subject Alternative Names (SANs) in the certificate. If a SAN is available and does not match the reference identifier, then the verification fails, and the channel is terminated. If there is no SAN, then the verification fails, and the channel is terminated. If the SAN exists and does not match, then the verification fails, and the channel is terminated. Otherwise, the reference identifier verification passes, and additional verification actions can proceed.

The comparison between SAN in the certificate and the expected identifier is crucial for certificate validation. The MDM Server and MDM Agent verify the certificate's SAN to ensure it matches the expected identifier, the fully qualified domain name (FQDN) of the remote network entity. Wildcards are accepted by the MDM Server only. The SAN can be an exact match or a match with a wildcard, as per RFC 6125. This comparison ensures that the certificate presented by the remote network entity is valid and authorized for communication, enhancing the security of TLS sessions between the MDM Server, MDM Agent, and external servers. Any discrepancies between the expected identifier and the information in the certificate are flagged, and TLS sessions are not established until validation is successful.

In EPMM, wildcard certificate handling differs between the MDM server and MDM agent. The MDM Agent enforces strict identity validation by requiring an exact match of the Common Name (CN), as the CN is locally generated along with UUID of the agent and used for deterministic trust evaluation hence, wildcard certificates are not supported. On the other hand, the MDM server, when functioning as a client can accept wildcard certificates via the Subject Alternative Name (SAN) field, provided the Target of Evaluation (TOE) is designed around Fully Qualified Domain Name (FQDN) resolution.

The TOE does not implement certificate pinning. Each TOE component only supports the trusted CA certificate(s) needed for verification of the trust chain and its own entity certificate.

The TLS client will transmit the Supported Elliptic Curves extension in the Client Hello message by default with support for the following NIST curves: secp256r1, secp384r1, and secp521r1. The remote endpoint server can choose to negotiate the elliptic curve from this set for any of the mutually negotiable elliptic curve ciphersuites.

Refer to Section 6.1.10 FAU_STG_EXT.1 External Trail Storage *(MDM_PP)* for a description of the three TLS Client connections supported by the MDM Server. Refer to Section 6.4.2 Enrollment of Mobile Device into Management *(MDM_PP)* for a description of the two TLS Client connections supported by the MDM Agent.

6.3.16 FCS_TLSC_EXT.2 TLS Client Support for Mutual Authentication *(TLS_PKG)*

The MDM Server component of the TOE supports TLS Client mutual authentication when communicating with the Audit Server. Communicating with an Audit Server is the only Inter-TSF Trusted Channel, FPT_ITC.1(1), the TSF supports. Refer to Section 6.1.10 FAU_STG_EXT.1 External Trail Storage for a description of the three MA TLS Client channels implemented by the MDM Server. For these channels, the MDM Server must support mutual authentication however, mutual authentication is not required.

The MDM Agent component of the TOE supports one mutual authentication TLS Client trusted channel that is used to transfer data securely once enrolled (Line B Figure 1). The initial enrollment channel does not support mutual authentication. The second channel, used to transfer all data, requires mutual authentication. The certificate sent to the MDM Server on the second channel is the certificate received from the server during enrollment (6.4.2 Enrollment of Mobile Device into Management (MDM_PP)).

There are no factors beyond configuration that are necessary in order for the three client implementation to engage In mutual authentication using X.509 certificates.

6.3.17 FCS_TLSC_EXT.5 TLS Client Support for Supported Groups Extensions *(TLS_PKG)*

The product shall present the Supported Groups Extension in the Client Hello with the supported groups secp256r1, secp384r1, and secp521r1. support mutual authentication using X.509v3 certificates

6.3.18 FCS_TLSS_EXT.1 TLS Server Protocol *(TLS_PKG)*

The Ivanti EPMM TOE supports the TLS protocol (TLS 1.2 RFC 5246) for the purpose of protecting data in transit. Both the MDM Server component of the TOE and the MDM Agent component support TLSC functionality. Refer to Table 23: EPMM TLS Protocol Implementation for the itemized list of TLSC support.

The implementation of the TLS protocol in the MDM Server and MDM Agent components ensures that the cipher suites supported are explicitly specified. The supported cipher suites for both acting as a TLS server and as a TLS client include the following suites:

- TLS_RSA_WITH_AES_128_GCM_SHA256 as defined in RFC 5288,
- TLS_RSA_WITH_AES_256_GCM_SHA384 as defined in RFC 5288,
- TLS_ECDHE_ECDSA_WITH_AES_128_GCM_SHA256 as defined in RFC 5289,
- TLS_ECDHE_ECDSA_WITH_AES_256_GCM_SHA384 as defined in RFC 5289
- TLS_ECDHE_RSA_WITH_AES_128_GCM_SHA256 as defined in RFC 5289,
- TLS_ECDHE_RSA_WITH_AES_256_GCM_SHA384 as defined in RFC 5289

The MDM Server component explicitly denies old SSL and TLS versions. The MDM Server supports only TLS 1.2 (as per RFC 5246) for communication purposes. Any attempts to use older SSL or TLS versions are rejected by the components. This denial aligns with the requirements specified in the FCS_TLSS_EXT.1.2 requirement and ensures that outdated and potentially vulnerable protocols are not used..

The MDM Server generates key agreement parameters using NIST curves secp256r1, secp384r1, and secp521r1 when employing elliptic curve ciphers for the server's Key Exchange message and RSA key sizes of 2048, 3072, and 4096 bits for RSA key exchange. These key agreement parameters are dynamically generated during the TLS handshake process. The specific curve parameters correspond to the server certificate configured for client access. The selection of these curves ensures robust key agreement and cryptographic security for TLS sessions established between the MDM Server and clients, providing confidentiality and integrity for transmitted data.

6.3.19 FCS_TLSS_EXT.2 TLS Server Support for Mutual Authentication *(TLS_PKG)*

The MDM Server component of the TOE supports TLS mutual authentication using client-side certificates.

When a mobile device is enrolled, it is dynamically assigned a unique UUID that is associated with the serial number of the device's certificate. During the TLS handshake process, when the mobile device connects and presents its certificate, the UUID in the certificate is used to look up the certificate serial number to ensure it matches the certificate that was presented. This process ensures that only authenticated and authorized mobile devices can establish TLS sessions with the MDM Server. If an applicable certificate is found to be invalid or if the remote network entity doesn't match the SAN (Subject Alternative Name) in the certificate, TLS sessions will not be established. Any error messages resulting from certificate validation failures are provided prior to terminating the session, ensuring that administrators are informed of authentication issues.

The comparison between the SAN (Subject Alternative Name) in the certificate and the expected identifier is crucial for certificate validation. The MDM Server and MDM Agent verify the certificate's SAN to ensure it matches the expected identifier, the fully qualified domain name (FQDN) of the remote network entity. Wildcards are not accepted. This comparison ensures that the certificate presented by the remote

network entity is valid and authorized for communication, enhancing the security of TLS sessions between the MDM Server, MDM Agent, and external servers. Any discrepancies between the expected identifier and the information in the certificate are flagged, and TLS sessions are not established until validation is successful.

In EPMM, wildcard certificate handling differs between the MDM server and MDM agent. The MDM Agent enforces strict identity validation by requiring an exact match of the Common Name (CN), as the CN is locally generated along with UUID of the agent and used for deterministic trust evaluation hence, wildcard certificates are not supported. On the other hand, the MDM server, when functioning as a client can accept wildcard certificates via the Subject Alternative Name (SAN) field, provided the TOE is designed around Fully Qualified Domain Name (FQDN) resolution in SAN.

For user portal communication, the TOE validates the SAN otherName and rfc822Name.

The MDM Server can be configured for mutual authentication for the two Administrative interfaces, the Admin Portal and the System Manager but the CC evaluated configuration does not require MA on those interfaces.

Applied TD0770.

6.4 Identification and Authentication (FIA)

6.4.1 FIA_CLI_EXT.1 X.509 Unique Certificate *(MDM_PP)*

The MDM component of the TOE maps the client identity certificate to a corresponding record in the registration database. This is the method for verifying the identity of the client.

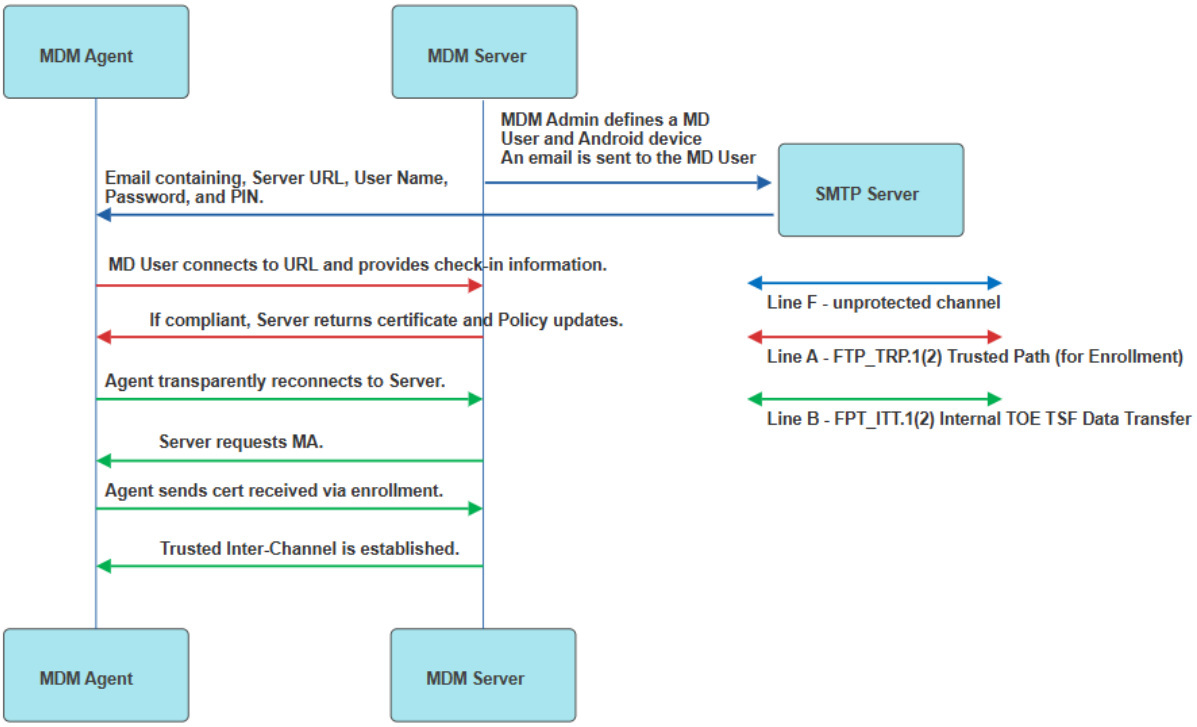
When a mobile device is enrolling (Enrollment TLS Trusted Path) it is assigned a unique UUID that is associated with the serial number of the device's certificate. When the mobile device connects and presents its certificate (MA Inter-TSF Trusted Channel), the UUID in the certificate is used to look up the certificate serial number to ensure it matches the certificate that was presented.

Note: This Mandatory SFR was added by TD0754 (the TD also deleted FIA_X509_EXT.5).

6.4.2 FIA_ENR_EXT.1 Enrollment of Mobile Device into Management *(MDM_PP)*

The following figure depicts the enrollment process.

Figure 3: Enrollment Process



To enable an MDM Agent to enroll, the MDM Server Administrator must configure Registration parameters that include username, password (using System Manager GUI), the Android device and the role of Mobile Device User.

Once the Administer selects “Register”, an email is sent (configured FAU_ALT_EXT.1 Server Alerts) to the Android user with the following information:

- MDM Server FQDN
- User name
- Password
- Registration PIN: a configurable length PIN valid for a configurable number of hours

When enrolling a new device, the initial connection is made via TLS with a secure connection using TLS during which only the MDM Server's certificate is verified by the MDM Agent (Line A Figure 1 FTP_TRP.1(2) Trusted Path (for Enrollment) *(MDM_PP)).* After establishing the TLS connection, the MD user must authenticate using a PIN and username/password that is configured on the MDM Server and sent to the user via the Server Alert (email).

The device can only be enrolled if the MDM Server approves it, ensuring it meets requirements i.e. not exceeding the maximum number of enrolled devices and meeting any device restrictions. Once approved, the MDM Server’s local CA (Certificate Authority) creates a new certificate and the MDM Server then returns the certificate to the MDM Agent. The Per-User Device Limit value is set using the Admin Portal Settings > Users & Devices > Registration parameter. Parameters like Policies are sent in the update bundle to MDM Agents when the Android devices check-in.

During enrollment the mobile device is assigned a unique UUID that is associated with the serial number of the device’s certificate. When the mobile device connects and presents its certificate (MA Inter-TSF

Trusted Channel), the UUID in the certificate is used to look up the certificate serial number to ensure it matches the certificate that was presented.

During enrollment and later synchronization, the MDM Server checks the MDM Agent's OS version and may quarantine it if it doesn't meet configured restrictions. Restrictions can be implemented by the system administrator during user's enrollment of devices by configuring specific "user credentials" and defining a limit on the number of devices that can be enrolled on the EPMM server. Additionally, a registration PIN and a passcode expiry (duration) may also be set by the administrator for initial authentication of the MD. Quarantine removes most enterprise configurations but allows check-in. If the MDM Agent complies during a later check-in, quarantined access will be restored.

When enrolling, the MDM Agent saves the MDM Server's unique URL (FQDN) for future communication. This URL is set by the mobile device user during the enrollment process.

6.4.3 FIA_ENR_EXT.2 Agent Enrollment of Mobile Device into Management *(AGT_PPM)*

The reference identifiers for TOE components or trusted IT entities are configured by the administrator using the available administrative commands when the certificates are created. The reference identifier for the MDM components in the x.509 certificates must be a DNS Name as described in RFC 6125 Section 6. This is true for both inter-TOE communications between TOE components and for communication with non-TOE trusted IT entities. The TOE doesn't support IP addresses as reference identifiers.

6.4.4 FIA_UAU.1 Timing of Authentication *(MDM_PP)*

The TOE supports five user interfaces. Each interface requires identification and authentication and the TOE prevents any users performing any TSF mediated action before being successfully identified and authenticated. The following identifies the users of the TOE.

- Android Users
  - The TOE only supports local login by users of the Android devices. Initially, the Users login to the Android OS (Operational Environment). Local Android Users invoke the MDM Agent software by initiating the enrollment process which means they connect to the MDM Server.
- Android users initiating enrollment from the MDM Agent to the MDM Server (Line A Figure 1).
  - Communication is via TLS version 1.2 RFC 5246 and its HTTPS is compliant with RFC 2818. Mutual Authentication is not required.
  - This interface only supports Android users with MD Mass User and MD User roles.
  - Authentication to the Server is by using Username/Password and PIN received from a Server Alert.
- Android users completing enrollment to the MDM Server (Line B Figure 1).
  - Communication is via TLS version 1.2 RFC 5246 and its HTTPS is compliant with RFC 2818. Mutual Authentication is required.
  - This interface only supports Android users with MD Mass User and MD User roles.
  - Authentication to the MDM Server is by using the X.509 certificate received from the initial enrollment channel sent as a mutual authentication request from the MDM Server.
- Remote Administrators logging into the Admin Portal (Line C Figure 1).
  - Communication is via TLS version 1.2 RFC 5246 and its HTTPS is compliant with RFC 2818. Mutual Authentication is not required.

- This interface only supports Administrators with Administrator and MAS Administrator roles.
- Authentication to the Server is by using Username/Password configured on the MDM Server.

- Remote Administrators logging into the System Manager (Line D Figure 1).
  - Communication is via TLS version 1.2 RFC 5246 and its HTTPS is compliant with RFC 2818. Mutual Authentication is not required.
  - This interface only supports Administrators with Administrator and MAS Administrator roles.
  - Authentication to the Server is by using Username/Password configured on the MDM Server.

6.4.5 FIA_X509_EXT.1(1) X.509 Certificate Validation *(MDM_PP)*

Both components of the TOE validate X.509 certificates. Certificates are validated when they are loaded onto a TOE system and when they are received from a remote system during the TLS handshake.

The MDM Server component of the TOE validates certificates of the servers it communicates with (Audit Server) when it is functioning as a TLS Client (Line E, Line F, and Line G Figure 1). The MDM Server component of the TOE also validates certificates of the clients it communicates with (MDM Agent) when it is functioning as a TLS Server (Line B Figure 1) on the mutual authenticated TLS channel. The MDM Server component of the TOE for both the Admin Portal and the System Manager do not support mutual authentication for the two GUI interfaces.

The MDM Agent component of the TOE validates the MDM Server's server certificates when functioning as a TLS Client both on the trusted path (Line A Figure 1 and Figure 2) and the inter-TSF trusted channel (Line B Figure 1 and Figure 2).

Certificate validation includes checking the validation path, basicConstraints, revocation, and extendedKeyUsage properties. Certificates must also chain to a trusted root that is configured by the administrator.

The TSF will treat a certificate as a Certificate Authority (CA) certificate only if the basicConstraints extension is present and the CA flag is set to TRUE.

The validation path is:

- MDM Server Component:
  - The MDM Server configures X.509v3 certificates for TLS authentication.
  - It performs certificate validation checks, including validation path, basicConstraints, revocation, and extendedKeyUsage properties.
  - TLS sessions will not be established if the applicable certificates cannot be determined to be valid.
  - Certificates must chain to a trusted root configured by the administrator.
  - The revocation checking is performed when certificates are uploaded to the TOE also during the establishment of a TLS connection supporting both CRL and OCSP. EPMM Server supports both CRL and OCSP when acting as a server with OCSP taking precedence. When EPMM Server is acting as a client, only CRL is supported.

- MDM Agent Component:
    - The MDM Agent uses X.509v3 certificates for TLS authentication.
    - It also performs validation checks, including validation path, basicConstraints, revocation, and extendedKeyUsage properties.
    - TLS sessions will not be established if the applicable certificate is not valid.
    - During the enrollment process, MDM Agents receive unique certificates, which they store in the Android keystore with permissions to only allow access by the applications themselves.
    - The revocation checking is performed when certificates are received during an enrollment session, also during the establishment of a TLS connection supporting both CRL and OCSP. EPMM Agent supports both CRL and OCSP with OCSP as priority.

Addressed TD0641

6.4.6 FIA_X509_EXT.2 X.509 Certificate Authentication *(MDM_PP)*

The TOE implements X.509 certificates for authentication of remote systems on a trusted channel. TLS Clients authenticate the remote TLS Servers and if the trusted channel supports mutual authentication (MA), the TLS Servers also authenticate the TLS Clients.

Refer to Table 23: EPMM TLS Protocol Implementation for the identification of the trusted channels supported by the TOE.

The TOE validates the server certificates when communicating to trusted Authorized IT Entities (FPT_ITC.1(1)). The only trusted IT entity defined is the Audit Server that supports both Syslog and HTTPS protocols (Line E, Line F and Line G Figure 1). Note that mutual authentication is not required for these channels, however, the TOE is required to support MA if the server requests it.

The MDM Server also authenticates the MDM Agent's certificate on the mutual authentication Internal TOE TSF Data Transfer trusted channel (FPT_ITT.1(2)) (Line B Figure 1).

The MDM Agents authenticate the MDM Server's certificates when connecting on the two trusted channels the TOE supports between the two components: the Trusted Path for Enrollment (FTP_TRP.1(2) Line A Figure 1) and the Internal TOE TSF Data Transfer trusted channel (FPT_ITT.1(2) Line B Figure 1)).

For all certificate authentication supported by the TOE, if the validation fails, the connection is not established.

The TOE maintains multiple X509 certificates namely Admin portal/System Manager certificate, mutual-authentication enrollment CA certificate, Audit server certificate and MDM Agent certificates. The TOE chooses an appropriate certificate based on the function that it performs. For example when an administrator connects to the MDM server, the TOE uses Admin portal certificate for authentication.

Additionally, the X.509 certificates are used to sign the policy bundles. Refer to Section 6.5.4 FMT_POL_EXT.1 Trusted Policy Update *(MDM_PP)* for a description of X.509 signed policy updates.

Both the MDM Server and MDM Agent components of the TOE implement the certificate authentication functions by including the cryptographic libraries used to validate certificates in the TOE's physical boundary. The MDM Servers use the Ivanti MDM OpenSSL Component cryptography library for all TLS paths/channels except the initial TLS connection to the Syslog Server. For that connection, Ivanti MDM Bouncy Castle cryptographic library is used. The MDM Agents use the Ivanti MDM Android Client OpenSSL

Component cryptography library. Both Android platforms perform identically. All three libraries are included in the TOE physical boundary and are CAVP certified as detailed in Table 28: CAVP Algorithm Certificate References .

6.5 Security Management (FMT)

6.5.1 FMT_MOF.1(1) Management of Functions Behaviour *(MDM_PP)*

The security management functions within the MDM Server component of the TOE are exclusively restricted to authorized administrators accessing the server via the Admin Portal or the Service Manager GUI interfaces. This restriction is implemented through role-based access controls, wherein each function is assigned specific access rights. Refer to Section 6.5.9 Security Management Roles *(MDM_PP)* for a description of management of roles. The following table identifies the management functions supported by MDM Server and the assigned role.

Table 24: EPMM MDM Server Management Functions and Roles

SFR	EPMM Server Mandatory Management Functions	Administrator Role	MAS Administrator Role
FMT_SMF.1.1(1) Specifications of Management Functions (Server configuration of Agent) *(MDM_PP)*	1. Transition to the locked state.	✓	
	2. Full wipe of protected data.	✓	
	3. Unenroll the Android device.	✓	
	4. Install policies on an Android Device.	✓	
	5. Query connectivity status of an Android device.	✓	
	6. Query the current version of the Android device's firmware/software.	✓	
	7. Query the current version of the Android device's hardware.	✓	
	8. Query the current version of the Android device's installed mobile applications.	✓	
	9. Import X.509v3 certificates into the Android device's Trust Anchor Database.	✓	
	10. Install an application on an Android device.	✓	
	11. Update the EPMM software on an Android device.	✓	
	12. Remove an application from an Android device.	✓	✓
	25. Update an Android device's Password Policy that includes specifying: • Minimum password length • Minimum password complexity and	✓	

SFR	EPMM Server Mandatory Management Functions	Administrator Role	MAS Administrator Role
	• Maximum password lifetime.		
	26. Update an Android device's Session Locking Policy that includes specifying: • Screen-lock enable/disable, • Screen-lock timeout, • Number of authentication attempts.	✓	
	27. Update an Android device's wireless network Policy that defines which networks an Android device may join.	✓	
	28. Update the Security Policy for each wireless network that a. specifies the FQDN of acceptable WLAN authentication server certificates. b. Specifies the security type. c. Specifies the authentication protocol. d. Specifies the client credentials to be used for authentication.	✓	
	29. Specify an application installation Policy (MAS) that specifies an authorized application repository(s).	✓	✓
	30. Update an Android device's enable/disable Policy for • Camera • microphone	✓	
FMT_SMF.1.1(1) Specifications of Management Functions (Server configuration of Agent) *(MDM_PP)*	Enable, disable, and modify policies listed in FMT_SMF.1(1).	✓	
FMT_SMF.1.1(2) Specifications of Management Functions (Server configuration of Server) *(MDM_PP)*	b. Configure devices by specifying ○ a unique device ID ○ a number of devices	✓	
FMT_SMF.1.1(1) Specifications of Management Functions (Server configuration of Agent) *(MDM_PP)*	Enable, disable, and modify policies listed in FMT_SMF.1(3).	✓	
	a) Configure application access groups	✓	✓

SFR	EPMM Server Mandatory Management Functions	Administrator Role	MAS Administrator Role
FMT_SMF.1.1(3) Specifications of Management Functions (MAS Server) *(MDM_PP)*	b) Download applications.	✓	✓

Applications and updates, which are treated similarly to policies, are packaged using XML, digitally signed, and securely transmitted from the MDM Server to the MDM Agent via HTTP over TLS.

Administrators can initiate applications or updates by creating configuration policies, which are then sent to the MDM Agent during its next check-in. Alternatively, the MDM Server can be configured with a list of available applications, allowing mobile device users to use the MDM Agent to query, select and install applications or updates from the list.

6.5.2 FMT_MOF.1(2) Management of functions Behaviour (Enrollment) *(MDM_PP)*

MDM Services prevent unauthorized users from enrolling by requiring registration and authentication of MD Users. The MD Server Administrator configures Android devices on the MDM Server and issues a 6-12 digit PIN or password to the mobile device user. This PIN or password is necessary for enrolling the device, ensuring that only authorized users can access the MDM services.

Refer to FIA_ENR_EXT.1 Enrollment of Mobile Device into Management *(MDM_PP)* above for a enrollment process description.

6.5.3 FMT_MOF.1(3) Management of Functions in (MAS Server Downloads) *(MDM_PP)*

The Ivanti EPMM TOE supports Mobile Application Store (MAS) functionality. MAS hosts host applications for the enterprise and securely transmits applications to enrolled MDM Agents during its next check-in.

Administrators can also configure the MDM Server with a list of available applications, enabling mobile device users to use the MDM Agent to query and install any available applications or updates applications from the list.

An MD User can initiate an application update or an MDM Administrator or MDM Server can initiate an update. Users must be assigned the role of Administrator or Application Access Group to support MAS functionality.

In support of application hosting, the MDM Server's MAS functionality supports the configuration of application groups in the form of labels assigned to individual apps and devices. Only devices (MDM Agents) may download apps if the app and device labels match.

6.5.4 FMT_POL_EXT.1 Trusted Policy Update *(MDM_PP) and* FMT_POL_EXT.2 Agent Trusted Policy Update *(AGT_PPM)*

Policies are distributed by the MDM Server to the MDM Agent on the mutually authenticated trusted TLS channel (FTP_ITC.1(2)) (Line B Figure 1).

When an MDM Agent checks in, the MDM Server returns a copy of the latest policies to the MDM Agent.

The MDM Server provides digitally signed policies and policy updates to the MDM Agent. Policies and policy updates (including policy settings, configurations, and applications) are formatted using XML and the entire XML structure is hashed using SHA-256. The hash is then signed by the MDM Server using the configured portal certificate sent to the MDM Agent when the mutual authenticated channel was established during enrollment (FCS_COP.1(3)) defined algorithms: RSA-2048 or ECDSA-P384) and the signed hash is added to the policy structure prior to being sent to any agents via a secure TLS channel.

When the MDM Agent receives the policy or policy update, it creates a SHA-256 hash of its contents (less the signed hash). Then it verifies that the calculated hash matches with the hash was signed using the private key of the same MDM server portal certificate that was used to receive the policy or policy update. The MDM Agent will only install policies and policy updates if both the hash matches and the hash was signed by the expected certificate. If the verification fails, the policy or policy update is discarded, an error notification is generated on the MDM Agent, and an alert is sent to the MDM Server, the agent will close the connection at the TCP layer, and will no longer send HTTP requests payloads to the server.

This process is repeated during the next check-In since the policy current on the MDM Server has not been applied.

The MDM Agent only accepts policies and policy updates that are digitally signed by the MDM Server. When the MDM Agent checks-in with the MDM Server, the MDM Server will send any policy or policy updates to the MDM Agent via the secure TLS channel if the policy or policy update has not yet been applied by the MDM Agent.

Policies are sent when a MDM Agent checks-in with a MDM Servers. Check-ins occur the MDM Agent's Sync Interval expires. This parameter is in the Sync Policy. The TOE, in the Common Criteria Evaluated Configuration, is configured to check-in with the MDM Server at least every eight hours.

Addressed TD0754.

6.5.5 FMT_SMF.1(1) Specification of Management Functions (Server configuration of Agent) *(MDM_PP)*

6.5.5.1 FMT_SMF.1(1) Detail

Parameters are sent to the MDM Agents by modifying MDM Server Policies, Configuration parameters, and Setting parameters.

The MDM Server component of the TOE supports the following commands sent to the MDM Agents.

- Transition to the locked state.

  Locking an Android device is performed by an administrator as performing the Action > Lock on a specific device from the Admin Portal. The command is sent immediately to the device. Upon receipt, the MDM Agent will lock the Android screen will appear and the login credentials will be required

- Full wipe of protected data.

Instructing an Android device to wipe (erase) protected data is performed by an administrator as performing the Action > Wipe on a specific device from the Admin Portal. The command is sent immediately to the device. Wiping a device is the same as unenrolling a device. Once wipe is initiated the device factory resets there by making the device to the preconfigured mode i.e. unenrolled.

- Unenroll from management.

  Instructing an Android device to wipe (erase) protected data is performed by an administrator as performing the Action > Wipe on a specific device from the Admin Portal. The command is sent immediately to the device. Wiping a device is the same as unenrolling a device. Once wipe is initiated the device factory resets there by making the device to the preconfigured mode i.e. unenrolled.

- Install policies.

  See below for policy updates.

- Query status
  - 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.

  The above information is always present and can be viewed in the GUI screens when an administrator logs onto the MDM Server because it is information exchanged during a device check-in. The information is as current as the value of the Sync Interval parameter in the device's Sync Policy. An administrator may select to view current information by selecting the Action > Force Device Check-In for a specific device from the Admin Portal. This will force a device check-in and update the above parameters as well as all new policies.

- Import X.509v3 certificates into the Trust Anchor Database.

- Install applications.

  The MDM Server supports MAS functionality. MAS functionality enables the MDM Server to download applications and then make those applications available to a group of MDM Agents to download to their device.

- Update system software.

  Refer to Section 6.6.6 FPT_TUD_EXT.1 Trusted Update *(MDM_PP)* for a description of Update system software.

- Remove applications and Enterprise applications.

The MDM Server supports the following policies on the mobile devices that enable the following parameters to be configured.

- Password Policy:
  - minimum password length
  - minimum password complexity
  - maximum password lifetime

The password parameters are updated by an administrator modifying fields in the Security Policy. The Administrator guidance provides detail information about the specific TSF Data being modified. When an Administrator modifies the Password Policy, it modifies the local copy of the MDM Servers policy. The new parameters will be sent to the MDM Agent when the MDM Agent next checks-in. Alternatively, if the Administrator wants to apply the policy immediately, the Administrator can select the Action > Force Device Check-In. This sends a message to the MDM Agent and forces the MDM Agent to update the new policies.

- Session locking Policy:
  - Screen-lock enabled/disabled.
  - Screen lock timeout
  - Configure the number of authentication failures.

  The session locking parameters are updated by an administrator modifying fields in the Security Policy and sending the new Policy update to the MDM Agent.
- Wireless networks (SSIDs) to which the MD may connect.

  The Wireless parameters are updated by an administrator modifying fields in the Wireless Network Policy. The Administrator guidance provides detail information about the specific TSF Data being modified. When an Administrator modifies the Wireless Network Policy, it modifies the local copy of the MDM Servers policy. The new parameters will be sent to the MDM Agent when the MDM Agent next checks-in. Alternatively, if the Administrator wants to apply the policy immediately, the Administrator can select the Action > Force Device Check-In. This sends a message to the MDM Agent and forces the MDM Agent to update the new policies.
- Security policy for each wireless network:
  - specify the CA(s) from which the MD will accept WLAN authentication server certificate(s)
  - ability to specify security type
  - ability to specify authentication protocol
  - specify the client credentials to be used for authentication

  The Wireless parameters are updated by an administrator modifying fields in the Wireless Network Policy. The Administrator guidance provides detail information about the specific TSF Data being modified. When an Administrator modifies the Wireless Network Policy, it modifies the local copy of the MDM Servers policy. The new parameters will be sent to the MDM Agent when the MDM Agent next checks-in. Alternatively, if the Administrator wants to apply the policy immediately, the Administrator can select the Action > Force Device Check-In. This sends a message to the MDM Agent and forces the MDM Agent to update the new policies.
- Application installation Policy by
  - specifying authorized application repository(s)

  The application repository is specified by an administrator modifying a Services parameter and enabling/disabling Google Play or Apps@Work or both. Upon selecting Apply, the new configuration will be sent to the MDM Agent.
- Enable/disable policy for the camera and microphone.

  The camera and microphone parameters are updated by an administrator modifying fields in the Lockdown Policy. The Administrator guidance provides detail information about the specific TSF Data being modified. When an Administrator modifies the Lockdown Policy, it modifies the local

copy of the MDM Servers policy. The new parameters will be sent to the MDM Agent when the MDM Agent next checks-in. Alternatively, if the Administrator wants to apply the policy immediately, the Administrator can select the Action > Force Device Check-In. This sends a message to the MDM Agent and forces the MDM Agent to update the new policies.

The table below lists all the management functions supported by the TOE and associated roles.

EPMM Server Mandatory Management Functions	Administrator Role	MAS Administrator Role
1. Transition to the locked state (MDF Function 6)	✓	
2. Full wipe of protected data (MDF Function 7)	✓	
3. Import X.509v3 certificates into the Trust Anchor Database (MDF Function 11)	✓	
4. Install applications (MDF Function 16)	✓	
5. Update system software (MDF Function 15)	✓	
6. Remove an applications (MDF Function 14)	✓	✓
25. Update an Android device's Password Policy that includes specifying: • Minimum password length • Minimum password complexity and • Maximum password lifetime (MDF Function 1)	✓	
26. Update an Android device's Session Locking Policy that includes specifying: • Screen-lock enable/disable, • Screen-lock timeout, • Number of authentication attempts (MDF Function 2)	✓	
27. Update an Android device's wireless network Policy that defines which networks an Android device may join (MDF Function 2)	✓	
28. Update the Security Policy for each wireless network that e. specifies the FQDN of acceptable WLAN authentication server certificates. f. Specifies the security type. g. Specifies the authentication protocol. h. Specifies the client credentials to be used for authentication. (WLAN Client Function 1)	✓	
29. Specify an application installation Policy (MAS) that specifies an authorized application repository(s).	✓	✓
30. Update an Android device's enable/disable Policy for • Camera • microphone	✓	

6.5.5.2 Mobile Device Fundamentals FMT_SMF.1 Mappings

The MDM Agent consists of a software application deployed on one of two Android mobile devices. Both MDM Agent devices have been Common Criteria evaluated:

- Galaxy S22 Ultra 5G Android 13
  Samsung Electronics Co., Ltd.
  Samsung Galaxy Devices on Android 13 – Spring, NIAP VID11342

- Galaxy S23 Ultra 5G Android 14
  Samsung Electronics Co., Ltd.
  Samsung Galaxy Devices on Android 14 – Spring, NIAP VID11444 (https://www.niap-ccevs.org/products/11444).

Android 14, VID11444 is still listed on the NIAP Product Compliant List. However, Android 13, VID11342 has been archived. There is not a current evaluated Android 13 evaluation that includes a Galaxy S22 Ultra 5G.

The following table maps the evaluated Android 14 FMT_SMF.1 claims and compares the list to the FMT_SMF.1 claims in this ST. The ST for the Android 14 VID11444 evaluation is *Samsung Electronics Co., Ltd. Samsung Galaxy Devices on Android 14 – Spring Security Target*, Version 0.5, April 24, 2024. The Conformance Claims for that evaluation are:

- *PP-Configuration for Mobile Device Fundamentals, Biometric enrollment and verification – for unlocking the device, Bluetooth, Virtual Private Network (VPN) Clients, and WLAN Clients*, Version 1.0, 24 October 2022 (CFG_MDF-BIO-BT-VPNC-WLANC_V1.0)
  - The PP-Configuration includes the following components:
    - Base-PP: *Protection Profile for Mobile Device Fundamentals*, Version 3.3, (PP_MDF_V3.3)
    - PP-Module: *PP-Module for Virtual Private Network (VPN) Clients*, Version 2.4, (MOD_VPNC_V2.4)
    - PP-Module: *PP-Module for Bluetooth*, Version 1.0, (MOD_BT_V1.0)
    - *PP-Module for WLAN Client*, Version 1.0 (MOD_WLANC_V1.0)
    - PP-Module: *collaborative PP-Module for Biometric enrolment and verification - for unlocking the device* - [BIOPP-Module], Version 1.1, September 12, 2022 (MOD_CPP_BIO_V1.1)
- Package Claims:
  - Functional Package: Functional Package for Transport Layer Security (TLS), Version 1.1, (PKG_TLS_V1.1" It provides a list of the functionality claimed in

Key:
Column Two:
M = Mandatory requirement for MDF
I = Implemented
N/C = Not Claimed
The bold options in column one indicate non-mandatory options.
The number preceding the listing in column three includes the number retained from the SFR.

Table 25: MDF to MDM FMT_SMF.1 Mapping

Android 14 ST FMT_SMF.1 Claims	M/I	EPMM ST FMT_SMF.1 Claims
1. configure password policy: a. minimum password length b. minimum password complexity c. maximum password lifetime	M	25. password policy: a. minimum password length b. minimum password complexity c. maximum password lifetime (MDF Function 1)
2. configure session locking policy: a. screen-lock enabled/disabled b. screen lock timeout c. number of authentication failures	M	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)
3. enable/disable the VPN protection: a. across device [b. on a per-app basis, c. on a per-group of applications processes basis]	M	N/C
4. enable/diable [NFC, Bluetooth, Wi-Fi, and cellular radios]	M	N/C
5. enabledisable [camera, microphone] a. across device [d no other method]	M	30. enable/disable policy for [*camera and microphone*] across device, [• <u>no other method</u>], (MDF Function 5)
6. transition to the locked state	M	1. transition to the locked state (MDF Function 6)
7. TSF wipe of protected data	M	2. full wipe of protected data (MDF Function 7)
8. Configure application installation by: [a. Restricting the sources of applications b. Specifying a set of allowed applications based on [application name, developer signature] (an application whitelist) c. Denying installation of applications]	M	29. application installation policy by [○ <u>specifying authorized application repository(s),</u>], (MDF Function 8)
9. Import keys/secrets into the secure key storage	M	N/C

Android 14 ST FMT_SMF.1 Claims	M/I	EPMM ST FMT_SMF.1 Claims
10. Destroy imported keys/secrets and [no other keys/secrets] in the secure key storage	M	N/C
11. Import X.509v3 certificates into the Trust Anchor Database	M	9. import X.509v3 certificates into the Trust Anchor Database (MDF Function 11)
12. Remove imported X.509v3 certificates and [default X.509V3 certificates] in the Trust Anchor Database		N/C
13. enroll the TOE in management	M	N/C
14. remove applications	M	12. remove applications (MDF Function 14)
15. update system software	M	11. update system software (MDF Function 15)
16. install applications	M	10. install applications (MDF Function 16)
17. remove Enterprise applications	M	N/C
18. enable/disable display notification in the locked state of: [f. all notifications]	M	N/C
19. enable data-at rest protection	M	N/C
20. enable removable media's data-at-rest protection	M	N/C
21. enable/disable location services: a. across device [d. no other method]	M	N/C
22. enable/disable the use of [Fingerprint Authentication Factor, Hybrid Authentication Factor]	I	N/C
23. configure whether to allow/disallow establishment of a trusted channel if the peer/server certificate is deemed invalid.	I	N/C
24. enable/disable all data signaling over [assignment: list of externally accessible hardware ports]	I	N/C
25. N/C		N/C
26. enable/disable developer modes	I	N/C
27. enable/disable bypass of local user authentication	I	N/C
28. wipe Enterprise data	I	N/C

Android 14 ST FMT_SMF.1 Claims	M/I	EPMM ST FMT_SMF.1 Claims
29. N/C		N/C
30. N/C		N/C
31. N/C		N/C
32. read audit logs kept by the TSF	I	N/C
33. N/C		N/C
34. N/C		N/C
35. N/C		N/C
36. configure the unlock banner	M	N/C
37. N/C		N/C
38. N/C		N/C
39. enable/disable [a. USB mass storage mode]	I	N/C
40. N/C		N/C
41. enable/disable [a. Hotspot functionality authenticated by [pre-shared key], b. USB tethering authenticated by [passcode]]	I	N/C
42. approve exceptions for sharing data between [groups of applications]	I	N/C
43. place applications into application process groups based on [creating a Knox Separated Apps folder]	I	N/C
44. unenroll the TOE from management	I	N/C
45. Enable/disable the Always On VPN protection a. across device [b. on a per-app basis, c. on a per-group of applications processes basis]]	I	N/C
46. N/C		N/C
47. additional management functions [• enable/disable USB host storage • disable CC Mode • enable/disable manual Date/Time changes	I	N/C

Android 14 ST FMT_SMF.1 Claims	M/I	EPMM ST FMT_SMF.1 Claims
• enable/disable applications (including pre-installed)]		

Applied TD0479.

6.5.6 FMT_SMF.1(2) Specification of Management Functions (Server configuration of Server) *(MDM_PP)*

MDM Servers may restrict the number of devices a user may be allowed to be enroll. This is via the Admin Portal Settings > User & Devices Registration parameter.

When an MDM Agent is enrolled, an X.509 certificate is created. The device is assigned a unique device ID that is created from the certificate's serial number field.

The MDM Servers do not collect any privacy-sensitive information from the enrolled mobile devices. (The ST does not claim function c.4.)

6.5.7 FMT_SMF.1(3) Specification of Management Functions (MAS Server) *(MDM_PP)*

The MDM Server, which also functions as a MAS (Mobile Application Store) server and supports the configuration of application groups by the admin. Additionally, it enables the downloading of applications for deployment.

Groups are implemented by the MDM Server as "Labels". The TOE has both default and custom labels that Administrator can create with specific criteria. Some of the default groups/labels include "All-Smartphones", "Android" and "Employee-Owned" etc. Admin can create custom labels both static and dynamic. The admin must manually assign these labels.

The MDM Server component of the TOE restricts all security management functions (identified below for FMT_SMF.1(1)/ FMT_SMF.1(2)/FMT_SMF.1(3)) to an authorized administrator. This is accomplished by role-based access controls (described below) assigned to each of the functions. Note that applications and application updates are generally handled just like policies – they are packaged using XML and digitally signed along with any other policy elements and sent securely from the MDM Server to the MDM Agent via secure TLS. Applications and updates can be initiated by the MDM Server by creating a configuration policy that includes the application or update – in this case the application will be sent to the MDM agent during its next check-in. Alternately, the MDM server can be configured with a list of available applications and a mobile device user can use the MDM agent to query that list to select and install any available applications or associated updates.

6.5.8 FMT_SMF_EXT.4 Specification of Management Functions *(AGT_PPM)*

The MDM Agent interacts with the Android platform to perform the following tasks:

- Import the certificates to be used to authenticate the Servers it communicates with over TLS channels.
- Configure an Android device to enable the device to join an EPMM System. Specifically, usernames/passwords are assigned to a device.
- Configure the time period a device is required to check-in with the MDM Server.

The method of use to support enrollment and configuration is the TOE supports two trusted channels that each support HTTP over TLS from the MDM Agents to the MDM Server. The MDM Server acts as the TLS Server and the MDM Agents act as the TLS Client. This remote enrollment and configuration is the only supported method for enrollment. Refer to Section 6.4.2 FIA_ENR_EXT.1 Enrollment of Mobile Device into Management *(MDM_PP)* for more information.

Both MDM Agents included in the TOE, Android 13 and Android 14, support the functions and policies.

6.5.9 FMT_SMR.1(1) Security Management Roles *(MDM_PP)*

The MDM Server component of the TOE supports permission-based roles. The permissions are fine-grained and include viewing and managing devices; applying and removing policies, configurations, and labels; viewing logs and events, and managing administrators and device spaces.

The permissions are grouped into the following categories.

- Device Management – includes permissions that enable an administrator to view and manage devices which include adding and deleting devices, applying and removing device labels, wiping devices, locking and unlocking devices, force a device check-in, push profiles to a device, and retire a device.
- Privacy Control - includes permissions that enable an administrator to view apps in device detail, View device IP and MAC addresses, and locate devices.
- Label Management - includes permissions that enable an administrator to view and manage labels assigned to devices.
- User Management - includes permissions that enable an administrator to view and manage users and user attributes.
- App Management - includes permissions that enable an administrator to view apps resident on the EPMM Server Host, distribute apps, import and edit apps, and manage app licenses.
- Configuration Management – view and manage configuration, apply and remove configuration label, and delete configuration.
- Policy Management - includes permissions that enable an administrator to view and manage policies, apply and remove the policy label, delete policies, and manage the compliance action policy.
- Compliance Policy Management – view and modify the Security Policy which contains compliance information.
- Logs and Events Management - includes permissions that enable an administrator to view audit logs, manage and view events, and manage and view certificates.
- Admin Management - includes permissions that enable an administrator to manage administrators ad device spaces.

The Common Criteria evaluated MDM Server includes four default user roles that are configured at TOE installation and configuration of the TOE into the CC evaluated configuration. The TOE, in the CC evaluated configuration, will have one administrator assigned to each of the following default roles.

- Administrator - This is the initial administrator that has full control of all functions. Administrators have comprehensive control over the MDM Server functions and have no restrictions except that Administrators cannot get into the root shell.

- MAS Administrator - Application Access Group – This role is assigned to administrators for the MAS functionality role required by FMT_SMR.1(2). It's a limited admin role that provides authority to manage the MAS functionality from the server. The TOE allows "Labels" to be defined and apps to be assigned. Devices can also be assigned to labels and the associated services to restrict or allow access to corresponding apps. A MAS Administrator has permission to define and assign Labels.
- MD User - This is a user added in the Admin Portal, but not assigned any administrative roles.
- MAS MD User - Application Access Group – This is the agent side of the FMT_SMR.1(2) requirement. This role is assigned to MD Users but add permission for MAS functionality.
- Enrolled Mobile Device – This is a mobile device that has been enrolled by the MDM Server when a MD User or a MAS MD User is successfully enrolled.

NSR = Non-Security Relevant

Table 26: EPMM Permission to Role Mappings

Security Relevant (SR)/Non-Security Relevant (NSR)	EPMM Server Permissions	Roles			
		Administrator	MAS Administrator	MAS MD User	MD User
Add device					
NSR	Manage ActiveSync device	✓			
NSR	Manage device enrollment (iOS only)	✓			
	Delete retired device	✓			
	Apply and remove device label	✓			
	Lock and unlock device	✓			
	Force device check-in	✓			
	Send message to device	✓			
	Change device ownership	✓			
	Edit custom device attribute	✓			
	Export to CSV	✓			
	Retire device	✓			
	Push profiles in device details	✓			
	Remove profiles in device details	✓			
NSR	Update Intune Compliance Status for devices	✓			
NSR	Manage AppTunnel	✓			
Policy Control					
	View apps and ibooks in device details	✓	✓	✓	
	View device IP and MAC address	✓	✓		

Security Relevant (SR)/Non-Security Relevant (NSR)	EPMM Server Permissions	Roles			
		Administrator	MAS Administrator	MAS MD User	MD User
	Locate device	✓	✓		
Label Management					
	View label	✓	✓		
	Manage label	✓	✓		
User Management					
	View user	✓	✓		
	Manage user	✓	✓		
	Edit custom user attribute values	✓	✓		
App Management					
	View app	✓	✓	✓	
	View app inventory	✓	✓	✓	
	View app dashboard	✓	✓	✓	
	Manage app	✓	✓		
	Distribute app	✓	✓		
	Import and edit app	✓	✓	✓	
	Manage reviews	✓	✓		
	Manage App Licenses	✓	✓		
Configuration Management					
	View configuration	✓	✓		
	Manage configuration	✓	✓		
	Apply and remove configuration label	✓	✓		
	Delete configuration	✓	✓		
Policy Management					
	View policy	✓	✓		
	Manage policy	✓	✓		
	Apply and remove policy label	✓	✓		
	Delete policy	✓	✓		
	Manage ActiveSync policy	✓			
	Manage compliance action	✓			
Compliance Policy Management					
	View compliance policy	✓			

Security Relevant (SR)/Non-Security Relevant (NSR)	EPMM Server Permissions	Roles			
		Administrator	MAS Administrator	MAS MD User	MD User
	Modify compliance policy	✓			
	Apply and remove compliance policy labels	✓			
Settings and Services Management					
	View settings and services	✓			
	Manage settings and services	✓			
	Manage custom attributes	✓			
Logs and Events Management					
	View Audit logs	✓	✓		
	View MDM activity	✓			
	View certificates	✓			
	View event settings	✓			
	View events	✓			
	Manage certificates	✓			
	Manage event settings	✓			
	Manage events	✓			
Admin Management					
	Manage administrators and device spaces	✓			
Content Management					
	View content, apply and remove content labels	✓			
	Manage Content	✓			
Ivanti Bridge for Windows 10					
NSR	Send script to Ivanti Bridge for Windows 10	N/A	N/A	N/A	
Apple Education Management					
NSR	View Apple Education	N/A	N/A	N/A	N/A
NSR	Manage Apple Education	N/A	N/A	N/A	N/A
NSR	Setup Apple Education	N/A	N/A	N/A	N/A
Ivanti Access					
NSR	Allows an administrator to configure and manage Ivanti Access Integration	✓			
Microsoft Graph					
NSR	Allows an administrator to view Microsoft Graph API settings used for Office 265 Apps protection	✓			

Security Relevant (SR)/Non-Security Relevant (NSR)	EPMM Server Permissions	Roles			
		Administrator	MAS Administrator	MAS MD User	MD User
NSR	Allows an administrator to edit Microsoft Graph API settings used for Office 265 Apps protection	✓			
NSR	Allows an administrator to send a Wipe command to a device or cancel an issued Wipe command before it is executed	✓			
Other Roles					
NSR	View device feature usage data	✓			
NSR	Common Platform Services (CPS)	✓			
NSR	Connector	✓			
NSR	API	✓			
NSR	Mobile App	✓			
NSR	Enforce single session (all spaces)	✓			
NSR	Migration	✓			

Refer to EPMM System CC Administrator Guidance for guidance instructing configuration of the TOE in the evaluated configuration.

6.5.10 FMT_SMR.1(2) Security Management Roles (MAS Server) *(MDM_PP)*

The MAS Server functionality of the MDM Server is configured and maintained by a unique user role, the Application Access Group.

- Application Access Group – The TOE allows "Labels" to be defined and apps to be assigned. Devices can also be assigned to labels and the associated services to restrict or allow access to corresponding apps. An Application Access Group user has the permission to define and assign Labels.
- Enrolled Mobile Device – This is a mobile device that has been enrolled by the MDM Server when a MD User or a MAS MD User is successfully enrolled.

Refer to Section 6.5.9 for a description of how the above roles are implemented in the TSF.

6.5.11 FMT_UNR_EXT.1 User Unenrollment Prevention *(AGT_PPM)*

Because the TOE's use case is [USE CASE 1] an enterprise-owned device for general purpose enterprise use. The TSF does not allow a Mobile Device User (MAS MD User or MD User role) to unenroll from the enterprise. The TSF does not provide the functionality that enables a user to unenroll. Additionally, Mobile Device Users enrolled and using the EPMM System are in the evaluated configuration supporting non-Android administrators. This non-administrator privilege status means the MAS MD User and MD User cannot uninstall or remove the MDM Agent Ivanti application.

6.6 Protection of the TSF (FPT)

6.6.1 FPT_API_EXT.1 Use of Supported Services and API's *(MDM_PP)*

The MDM Server component of the TOE is an application program running on an Oracle Linux OS. The MDM Server only uses well known, documented platform API's. The MDM Server application requires the following functionality which is provided by the operating system (Operational Environment).

- File System

  The MDM Server requires read, write, and file management capabilities to interact with files on the local device. This is used to manage syslog events, TSF data managed by the Apache GUI, and to store keys and certificates.

  The MDM Server stores keys and certificates in flat files that are assigned permissions in order to restrict those keys to the components/processes that use them.

  This functionality is provided by the Linux File System API.

- Network

  The MDM Server requires sending and receiving network packets as well as configuring specific network interfaces. The MDM Server's two cryptographic libraries interface to the network which includes the device drives.

  This functionality is provided by the OS' Network API.

- Time

  The MDM Server requires the current time and date and relies on the underlying platform to provide that information. Time is used when generating a syslog message and alerts.

  This functionality is provided by the OS' TimeAPI API.

- Package Manager

  The MDM Server uses the RPM Package Manager (RPM) functions to verify the integrity of its own executable files during start-up. The MDM Server also interfaces with this application to update the MDM Server software component of the TOE.

The MDM Agent component of the TOE is an application program running on Android 13 or Android 14. The MDM Agent only uses well known, documented platform API's. Both Android APIs are identical. The MDM Agent application requires the following functionality which is provided by the operating system (Operational Environment).

- File System

  The MDM Agent requires read, write, and file management capabilities to interact with files on the local device. This is used to manage audit events and TSF data and policies received from an MDM Server.

  This functionality is provided by Android's java.io package java library.

- Time of Day

  The MDM Agent requires the current time and date and relies on the underlying platform to provide the information. Time is used when generating audit records, validating certificates, and generating alerts.

  This functionality is provided by Android's java.time.chrono java library.

- Network

The MDM Agent requires sending and receiving network packets as well as configuring specific network interfaces. The MDM Agent's Ivanti MDM Android Client OpenSSL Component cryptographic library (included in the TOE) is the cryptographic library used to interface to the network.

This functionality is provided by the Android network app named NetworkStack.

- Keystore

  The MDM Agent receives X.509 certificates during their enrollment process and they store the received certificates in the Android Keystore. Keystore data Is encrypted using the Android Operational Environment's crypto library, Samsung BoringSSL Android 1.7 and 1.8 for Android 13 and 14 respectively.

  The MDM Agent uses the Android Keystore API to read and write certificates.

6.6.2 FPT_ITT.1(2) Internal TOE TSF Data Transfer (MDM Agent) *(MDM_PP)*

The TSF is a distributed TOE that includes one MDM Server and one or more MDM Agents.

The TOE MDM Server component implements a unique secure communication channel to communicate with each of the TOE's MDM Agent component (Line B Figure 1) . The MDM Server implements TLS Server functionality. The MDM Agent implements TLS Client functionality. The method of communication with the MDM Agent is by the TOE's mutually authenticated HTTP over TLS (HTTPS). The TOE implements version 1.2 of TLS and its HTTPS is compliant with RFC 2818.

During initial mobile enrollment (Line A Figure 1) only the MDM Server is authenticated by the MDM Agent (non-mutual authentication). The two components exchange information and if the MDM Server approves mobile enrollment for the device, the MDM Agent then reconnects to the MDM Server creating this trusted channel. Once enrolled, all communication between the MDM Server and the MDM Agent is protected using this mutual authentication TLS channel.

On the MDM Server, HTTP is provided by an Apache 2.4 library. The TCP/IP protocol stack, including TLS, is provided by the MDM OpenSSL Crypto Library. The network drivers are provided by the MDM Server's operating system (Oracle Linux 8.9). Apache and the OpenSSL Crypto Library are included in the TOE boundary. The network drivers are in the Operational Environment. The OpenSSL Crypto Library is CAVP certified as defined in Table 28: CAVP Algorithm Certificate References in this document.

On the MDM Agent, HTTP is provided by an Apache 2.4 library. The TCP/IP protocol stack, including TLS, is provided by the Android OpenSSL Crypto Library. The network drivers are provided by the MDM Agent's operating system (Android 13 or Android 14). Apache and the Android OpenSSL Crypto Library are included in the TOE boundary. The network drivers are in the Operational Environment. The Android OpenSSL Crypto Library is CAVP certified as defined in Table 28: CAVP Algorithm Certificate References in this document.

6.6.3 FPT_LIB_EXT.1/AGT Use of Third Party Libraries *(MDM_PP)*

The MDM Agent component of the TOE's software build includes the following libraries:

- Ivanti MDM Android Client OpenSSL Component Cryptographic Library, v2.2.1 and
- OTS libraries listed in Appendix B.

Addressed TD0895.

6.6.4 FPT_LIB_EXT.1/SRV Use of Third Party Libraries *(MDM_PP)*

The MDM Server component of the TOE's software build includes the following libraries:

- Apache 2.4,
- Ivanti MDM Bouncy Castle, v1.0.2.4,
- Ivanti MDM OpenSSL Component, 1.1.1g, and
- OTS libraries listed in Appendix B.

Addressed TD0895.

6.6.5 FPT_TST_EXT.1 Functionality Testing *(MDM_PP)*

The self-test are initiated during initial power-up of the TOE as well as whenever the administrator issues a reboot for the MDM Server.

The MDM Server component of the TOE relies on both the Operational Environment and the TOE to perform self-tests. The MDM Server uses the Operational Environment's RPM Package Manager (RPM) functions of the TOE to verify the integrity of its own executable files. Additionally, the MDM Server's FIPS certified cryptographic modules (Ivanti MDM Bouncy Castle and Ivanti MDM OpenSSL Component) include self-tests to ensure the available cryptographic operations are performing correctly when starting up. Any errors are reported on the VMWare ESXi console and the MDM Server will attempt to restart if an error is detected without intervention by an Administrator.

At system start-up, the MDM Server uses the Operational Environment's RPM Package Manager (RPM) functions of the TOE to verify the integrity of its own executable files. The RPM Package Manager maintains a database of all installed RPMs, including the TSF, and during boot each RPM is checked for integrity with the standard Linux (Oracle) RPM integrity checking functions. Any errors are reported on the console and the MDM Server will attempt to restart if an error is detected without intervention by an Administrator. This RPM integrity test is sufficient to guarantee the integrity of the stored TOE executable code has not been compromised.

Once the RPM integrity tests pass and the system has entered into the operational state, the TSF performs self-tests on all approved cryptographic algorithms. This includes both the Ivanti MDM Bouncy Castle crypto library and the Ivanti MDM OpenSSL Component crypto library. The tests run, include but are not limited to, the algorithms identified in Table 28: CAVP Algorithm Certificate References. During these tests, services are not available, and data output is inhibited during the conditional self-tests. Specifically, the tests run by each cryptographic library are:

- Known Answer Tests – These tests verify the correct operation of cryptographic algorithms before they can be used in FIPS mode. They execute predefined tests vectors and compare outputs against known correct values.
- Power-on Self-Test (POST) – This test runs automatically when and ensures that the crypto library is ready for FIPS compliance.
- Hash Function Self-tests – These tests support multiple SHA variants using a unified testing framework.
- Algorithm Size Calculation Functions – Each algorithm implements size calculation functions to determine memory requirements. These tests help ensure that the algorithms function correctly and are not susceptible to attacks.

If any of these tests fail, an error message with be displayed on the console, the system will transition to the Error State, and halts processing. If this occurs, the Administrator should contact Ivanti support.

Otherwise, if the tests are successful, the system transitions into the operational state. Successful completion of the self-tests and transitioning into the operational state demonstrates that the TSF is operating correctly.

Refer to the Administrator guidance for more details and for instructions an Administrator is to take upon failure of self-tests.

Note: TD0438 removed the requirement for Agents in a distributed TOE to perform self-tests.

6.6.6 FPT_TUD_EXT.1 Trusted Update *(MDM_PP)*

6.6.6.1 Updating MDM Server TOE Component

The System Manager portal of the MDM Server component is used to query the version of the MDM Server that is installed. Administrators navigate to the Admin Portal Maintenance > Software Updates page to view the current version of the build.

The MDM Server component of the TOE enables Administrators to update the MDM Server software component. The System Manager portal interface is used for the MDM Server update communicating.

The MDM Server, Ivanti Endpoint Manager Mobile (Core), can be download by customers from the Ivanti site https://forums.ivanti.com/s/contactsupport?language=en_US. If required, the software must be moved to a host (the MDM Server Host Server Table 2) in order to be accessible by the TOE. The Administrator then uses the Software Updates option available by the System Manager's GUI to download the MDM Server component of the TOE.

RPMs are signed with an Ivanti signing key which is an RSA 2048-bit SHA-256 key. When a package is installed, the signature on the package is validated by the RPM tool. The RPM includes digests of the files within the RPM. These digests are stored in a database on the system during package install. During TOE upgrade, the contents of each file are verified against the stored digests.

To download, the Administrator selects the System Manager's Download option. Once downloaded, the Administrator must select Validate, to validate the digital signature. The Validate option will validate the software using the RPM tool (Operational Environment). A message will be displayed reporting the success or failure of the signature.

If the validation was successful, the Administrator must select the "Stage for Install" option to apply the MDM Server software. Only successfully signed packages will be allowed to be installed. Once the "Stage for Install" option is selected, the system is rebooted.

6.6.6.2 Updating MDM Agent TOE Component

MD Users may find the version of their MDM Agent component of the TOE (Ivanti MDM Agent for Android) installed by viewing the app's About Screen from their Android device.

MDM Server administrators are contacted by Ivanti when a new MDM Agent build is available. The agent's build is a unique build of MDM Agent software used for the Common Criteria evaluation and is not the same build that is available on Google Play Store or the Ivanti website. The evaluated MDM Agent build is referred to by Ivanti as the Federal Build.

Once notified, the MDM Server administrators download the new version of the build on a host, accessible by the Android users and notify each MDM Agent by sending an email (FAU_ALT_EXT.1 Server Alerts). The email includes the URL MD Users are to use to download the build and the version number. MD Users

can compare their running version to the version received to determine if they are running the current version.

All releases are packaged in APK (Android Package) format and are signed with an Ivanti software certificate.

To install the application, the Android User issues the adb install <downloaded .apk file> (Android Debug Bridge) command. Success or failure of the operation will be displayed on the screen. The upgrade is successfully performed only if the digital signature of the new build matches the digital signature of the current build.

MDM Users are encouraged to upgrade their version immediately because once the new version is made available, the MDM Server Administrator marks this version as valid and upon next check-in, when the device sends its configuration information, the Android device will become non-compliant if the upgrade was not applied (FAU_CRP_EXT.1).

6.7 Trusted Path/Channels (FTP)

6.7.1 FTP_ITC.1(1) Inter-TSF Trusted Channel (Authorized IT Entities) *(MDM_PP)*

The TOE communicates with one authorized IT entity, an Audit Server. MDM Server component implements three secure communication channels to communicate with the Audit Server.

The method of communication with the authorized IT entity is by the TOE's mutually authenticated TLS version 1.2 (RFC 5246) implementation. The MDM Server acts as a TLS Client communicating to the Audit Server, providing TLS Server functionality.

The MDM Agent component of the TOE does not initiate communication with any Authorized IT Entities as defined in the PP_MDM_V4.0.

An Audit Server is required to be configured for the TOE in the CC evaluated configuration. Once configured, the TOE, using the Bouncy Castle Cryptographic Library (in the TOE boundary), connects to the Audit Server (Line E Figure 1). If the server's X.509 certificate is successfully validated, the MDM Server then connects again, using the MDM OpenSSL Crypto Library (in the TOE boundary). This is the channel (Line F Figure 1) used to send all audit records. If this channel is interrupted, the TOE will attempt to reconnect this connection.

Audit Records created by Admin Portal actions are automatically sent to the Syslog Server port (configurable). Audit Records generated by the MDM Agent are pulled from the devices from the Admin Portal by an Administrator by selecting a device and then selecting the Action of Pull Client Logs. Once resident on the MDM Server host, an Administrator can transfer the records to the Audit Server by selecting Export Logs option from the System Manager. The protocol used to transfer Client audit records is HTTP over TLS and compliant to RFC 2818 Figure 1 (Line G Figure 1). The Audit Server, configured as an HTTPS Server, is required by the CC Evaluated Configuration and configured when the TOE is installed.

HTTP is provided by an Apache 2.4 library. Syslog, the TCP/IP protocol stack, including TLS, is provided by the Ivanti MDM OpenSSL Component Crypto Library for the second syslog connection and the HTTPS connection to the Audit Server. The Ivanti MDM Bouncy Castle crypto library provides the crypto library for the first connection to the Audit Server. The network drivers are provided by the MDM Server's operating system (Oracle Linux 8.9). Apache and the OpenSSL Crypto Library are included in the TOE boundary. The network drivers are in the Operational Environment. The OpenSSL Crypto Library is CAVP certified as detailed in Table 28: CAVP Algorithm Certificate References.

6.7.2 FTP_ITC_EXT.1 Trusted Channel *(MDM_PP)*

The TOE is a distributed TOE that includes a host running the MDM Server application and one or more Android devices running the MDM Agent application. Both the MDM Server software component and the MDM Agent software are included in this evaluation and therefore, communication between the two TOE components is considered internal communication to the TOE.

The TOE includes MAS Server functionality. This functionality is included in the MDM Server application and therefore does not provide additional distributed functionality.

6.7.3 FTP_TRP.1(1) Trusted Path (for Remote Administration) (MDM_PP)

The MDM Server offers two GUI interfaces used for remote administration: the Admin Portal and the System Manager. The MDM Server listens on the standard TCP HTTPS port 433 for the Admin Portal and listens on port 8443 for System Manager connections. In general, the Admin Portal provides management functions used for day-to-day management such as viewing the status of the Android devices and updating configuration and policies. The System Manager includes functionality that typically occurs at installation.

Remote Administrators can initiate each web-based session, secured through using HTTP over TLS (HTTPS). The TOE provides HTTPS Server functionality and the remote workstations are the HTTPS Clients. These trusted paths are identified as Lines C and D Figure 1.

HTTP is provided by an Apache 2.4 library. The TCP/IP protocol stack, including TLS, is provided by the MDM OpenSSL Crypto Library. The network drivers are provided by the MDM Server's operating system (Oracle Linux 8.9). Apache and the OpenSSL Crypto Library are included in the TOE boundary. The network drivers are in the Operational Environment. The OpenSSL Crypto Library is CAVP certified as defined in Table 28: CAVP Algorithm Certificate References in this document.

The MDM Agents do not provide a Trusted Path for Remote Administration. All interaction is via a local interface.

6.7.4 FTP_TRP.1(2) Trusted Path (for Enrollment) *(MDM_PP)*

The MDM Server's Trusted Path for Enrollment is the first channel initiated by the MDM Agent to enroll in the MDM Management (Line A Figure 1). Refer to Section 6.4.2 FIA_ENR_EXT.1 Enrollment of Mobile Device into Management *(MDM_PP)* for a description of enrollment.

On the MDM Server, HTTP is provided by an Apache 2.4 library. The TCP/IP protocol stack, including TLS, is provided by the MDM OpenSSL Crypto Library. The network drivers are provided by the MDM Server's operating system (Oracle Linux 8.9). Apache and the OpenSSL Crypto Library are included in the TOE boundary. The network drivers are in the Operational Environment. The OpenSSL Crypto Library is CAVP certified as defined in Table 28: CAVP Algorithm Certificate References in this document.

On the MDM Agent, HTTP is provided by an Apache 2.4 library. The TCP/IP protocol stack, including TLS, is provided by the Android OpenSSL Crypto Library. The network drivers are provided by the MDM Agent's operating system (Android 13 or Android 14). Apache and the Android OpenSSL Crypto Library are included in the TOE boundary. The network drivers are in the Operational Environment. The Android OpenSSL Crypto Library is CAVP certified as defined in Table 28: CAVP Algorithm Certificate References in this document.

6.8 EPMM MDM Cryptography

6.8.1 EPMM MDM Cryptographic Libraries

The three crypto libraries included in the Ivanti EPMM TOE are described in the following tables below.

Table 27: Ivanti EPMM Cryptographic Library CAVP Details

ST Reference	CAVP #cert Implementation Name	Runs On	CAVP #cert Operating Environment	CAVP #cert Description	CAVP #cert Version	CAVP #cert Vendor	CAVP #cert
Android OpenSSL	Ivanti MDM Android Client OpenSSL Component	MDM Agent Galaxy S22 Ultra 5G Android 13	Android 13 running Samsung Exynos 2200 without PAA	Android OpenSSL component, used in Ivanti MDM Android client application.	2.2.1	Ivanti, Inc.	#A6402
		MDM Agent Galaxy S22 Ultra 5G Android 14	Android 14 running Qualcomm Snapdragon 8 Gen 2 without PAA				
Bouncy Castle	Ivanti MDM Bouncy Castle Component	MDM Server Intel(R) Xeon(R) Gold 5215 CPU (Cascade Lake)	Oracle Linux 8.9 on VMWare ESXi 7.02 with Intel Xeon Gold 5215 without PAA	Bouncy Castle (1.0.2.4) cryptographic library operating in Java SE Runtime Environment 8	1.0.2.4	Ivanti, Inc.	#A6073
MDM OpenSSL	Ivanti MDM OpenSSL Component		Oracle Linux 8.9 on VMWare ESXi 7.02 with Intel Xeon Gold 5215 without PAA	Oracle Linux OpenSSL Module (openssl-libs-1.1.1g-15.el8_3.x86_64)	1.1.1g	Ivanti, Inc.	#A6074

6.8.2 CAVP Algorithm Certificate Details

The following table identifies the CAVP certified cryptographic libraries included in the TOE boundary.

Table 28: CAVP Algorithm Certificate References

Platform Server or Agent	CAVP Algorithm	Standard	Modes Supported	Crypto Library	Cert #
FCS_CKM.1 Cryptographic Key Generation *(MDM_PP)*					

Platform Server or Agent	CAVP Algorithm	Standard	Modes Supported	Crypto Library	Cert #
MDM Agent	RSA KeyGen (FIPS186-5)	RSA schemes that meet FIPS PUB 186-5, “Digital Signature Standard (DSS)”, Appendix B.3.	2048, 3072, and 4096 bits	Ivanti MDM Android Client OpenSSL Component 2.2.1	#A6402
MDM Agent	ECDSA KeyGen (FIPS186-5)	ECC schemes that meet FIPS PUB 186-5, “Digital Signature Standard (DSS)”, Appendix B.4.	P-256, P-384, and P-521	Ivanti MDM Android Client OpenSSL Component 2.2.1	#A6402
MDM Server	RSA KeyGen (FIPS186-5)	RSA schemes that meet FIPS PUB 186-5, “Digital Signature Standard (DSS)”, Appendix B.3.	2048, 3072, and 4096 bits	Ivanti MDM OpenSSL Component 1.1.1g	#A6074
MDM Server	ECDSA KeyGen (FIPS186-5)	ECC schemes that meet FIPS PUB 186-5, “Digital Signature Standard (DSS)”, Appendix B.4.	P-256, P-384, and P-521	Ivanti MDM OpenSSL Component 1.1.1g	#A6074
MDM Server	RSA KeyGen (FIPS186-5)	RSA schemes that meet FIPS PUB 186-5, “Digital Signature Standard (DSS)”, Appendix B.3.	2048, 3072, and 4096 bits	Ivanti MDM Bouncy Castle 1.0.2.4	#A6073
MDM Server	ECDSA KeyGen (FIPS186-5)	ECC schemes that meet FIPS PUB 186-5, “Digital Signature Standard (DSS)”, Appendix B.4.	P-256, P-384, and P-521	Ivanti MDM Bouncy Castle 1.0.2.4	#A6073
FCS_CKM.2 Cryptographic Key Establishment *(MDM_PP)*					
MDM Agent	Lab tested using known good implementation	RSA-based key establishment schemes that meet 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"	----	Ivanti MDM Android Client OpenSSL Component 2.2.1	No NIST CAVP, CCTL has performed all assurance/evaluation activities and documented in the ETR and AAR accordingly.

Platform Server or Agent	CAVP Algorithm	Standard	Modes Supported	Crypto Library	Cert #
MDM Agent	KAS-ECC-SCC Sp800-56Ar3	Elliptic curve-based key establishment schemes that meet the following: NIST Special Publication 800-56A Revision 2, “Recommendation for Pair-Wise Key Establishment Schemes Using Discrete Logarithm Cryptography”.	----	Ivanti MDM Android Client OpenSSL Component 2.2.1	#A6402
MDM Server	Lab tested using known good implementation	RSA-based key establishment schemes that meet 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"	----	Ivanti MDM Bouncy Castle 1.0.2.4	No NIST CAVP, CCTL has performed all assurance/evaluation activities and documented in the ETR and AAR accordingly.
MDM Server	KAS-ECC-SCC Sp800-56Ar3	Elliptic curve-based key establishment schemes that meet the following: NIST Special Publication 800-56A, “Recommendation for Pair-Wise Key Establishment Schemes Using Discrete Logarithm Cryptography”.	----	Ivanti MDM Bouncy Castle 1.0.2.4	#A6073
MDM Server	Lab tested using known good implementation	RSA-based key establishment schemes that meet 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"	----	Ivanti MDM OpenSSL Component 1.1.1g	No NIST CAVP, CCTL has performed all assurance/evaluation activities and documented in the ETR and AAR

Platform Server or Agent	CAVP Algorithm	Standard	Modes Supported	Crypto Library	Cert #
					accordingly.
MDM Server	KAS-ECC-SCC Sp800-56Ar3	Elliptic curve-based key establishment schemes that meet the following: NIST Special Publication 800-56A Revision 2, "Recommendation for Pair-Wise Key Establishment Schemes Using Discrete Logarithm Cryptography".	----	Ivanti MDM OpenSSL Component 1.1.1g	#A6074
FCS_COP.1(1) Cryptographic Operation (Confidentiality Algorithms) *(MDM_PP)*					
MDM Agent	AES-GCM	NIST SP 800-38D	128-bit, 256-bit key length	Ivanti MDM Android Client OpenSSL Component 2.2.1	#A6402
MDM Server	AES-GCM	NIST SP 800-38D	128-bit, 256-bit key length	Ivanti MDM Bouncy Castle 1.0.2.4	#A6073
MDM Server	AES-GCM	NIST SP 800-38D	128-bit, 256-bit key length	Ivanti MDM OpenSSL Component 1.1.1g	#A6074
FCS_COP.1(2) Cryptographic Operation (Hashing Algorithms) *(MDM_PP)*					
MDM Agent	SHA-256	FIPS Pub 180-4	----	Ivanti MDM Android Client OpenSSL Component 2.2.1	#A6402
MDM Agent	SHA-384	FIPS Pub 180-4	----	Ivanti MDM Android Client OpenSSL Component 2.2.1	#A6402
MDM Agent	SHA-512	FIPS Pub 180-4	----	Ivanti MDM Android Client OpenSSL Component 2.2.1	#A6402
MDM Server	SHA-256	FIPS Pub 180-4	----	Ivanti MDM Bouncy Castle 1.0.2.4	#A6073

Platform Server or Agent	CAVP Algorithm	Standard	Modes Supported	Crypto Library	Cert #
MDM Server	SHA-384	FIPS Pub 180-4	----	Ivanti MDM Bouncy Castle 1.0.2.4	#A6073
MDM Server	SHA-512	FIPS Pub 180-4	----	Ivanti MDM Bouncy Castle 1.0.2.4	#A6073
MDM Server	SHA-256	FIPS Pub 180-4	----	Ivanti MDM OpenSSL Component 1.1.1g	#A6074
MDM Server	SHA-384	FIPS Pub 180-4	----	Ivanti MDM OpenSSL Component 1.1.1g	#A6074
MDM Server	SHA-512	FIPS Pub 180-4	----	Ivanti MDM OpenSSL Component 1.1.1g	#A6074
FCS_COP.1(3) Cryptographic Operation (Signature Algorithms) *(MDM_PP)*					
MDM Agent	RSA SigGen (FIPS186-5)	FIPS PUB 186-5, "Digital Signature Standard (DSS)", Section 4.	2048, 3072, and 4096 bits	Ivanti MDM Android Client OpenSSL Component 2.2.1	#A6402
MDM Agent	RSA SigVer (FIPS186-5)	FIPS PUB 186-5, "Digital Signature Standard (DSS)", Section 4.	2048, 3072, and 4096 bits	Ivanti MDM Android Client OpenSSL Component 2.2.1	#A6402
MDM Agent	ECDSA SigGen (FIPS186-5)	FIPS PUB 186-5, "Digital Signature Standard (DSS)", Section 5.	P-256, P-384, P-521	Ivanti MDM Android Client OpenSSL Component 2.2.1	#A6402
MDM Agent	ECDSA SigVer (FIPS186-5)	FIPS PUB 186-5, "Digital Signature Standard (DSS)", Section 5.	P-256, P-384, P-521	Ivanti MDM Android Client OpenSSL Component 2.2.1	#A6402
MDM Server	RSA SigGen (FIPS186-5)	FIPS PUB 186-5, "Digital Signature Standard (DSS)", Section 4.	2048, 3072, and 4096 bits	Ivanti MDM Bouncy Castle 1.0.2.4	#A6073

Platform Server or Agent	CAVP Algorithm	Standard	Modes Supported	Crypto Library	Cert #
MDM Server	RSA SigVer (FIPS186-5)	FIPS PUB 186-5, "Digital Signature Standard (DSS)", Section 4.	2048, 3072, and 4096 bits	Ivanti MDM Bouncy Castle 1.0.2.4	#A6073
MDM Server	ECDSA SigGen (FIPS186-5)	FIPS PUB 186-5, "Digital Signature Standard (DSS)", Section 5.	P-256, P-384, P-521	Ivanti MDM Bouncy Castle 1.0.2.4	#A6073
MDM Server	ECDSA SigVer (FIPS186-5)	FIPS PUB 186-5, "Digital Signature Standard (DSS)", Section 5.	P-256, P-384, P-521	Ivanti MDM Bouncy Castle 1.0.2.4	#A6073
MDM Server	RSA SigGen (FIPS186-5)	FIPS PUB 186-5, "Digital Signature Standard (DSS)", Section 4.	2048, 3072, and 4096 bits	Ivanti MDM OpenSSL Component 1.1.1g	#A6074
MDM Server	RSA SigVer (FIPS186-5)	FIPS PUB 186-5, "Digital Signature Standard (DSS)", Section 4.	2048, 3072, and 4096 bits	Ivanti MDM OpenSSL Component 1.1.1g	#A6074
MDM Server	ECDSA SigGen (FIPS186-5)	FIPS PUB 186-5, "Digital Signature Standard (DSS)", Section 5.	P-256, P-384, P-521	Ivanti MDM OpenSSL Component 1.1.1g	#A6074
MDM Server	ECDSA SigVer (FIPS186-5)	FIPS PUB 186-5, "Digital Signature Standard (DSS)", Section 5.	P-256, P-384, P-521	Ivanti MDM OpenSSL Component 1.1.1g	#A6074
FCS_COP.1(4) Cryptographic Operation (Keyed-Hash Message Authentication) *(MDM_PP)*					
MDM Agent	HMAC-SHA2-256	Keyed-hash message authentication that meet the following: FIPS Pub 198-1, "The Keyed-Hash Message Authentication Code, and FIPS Pub 180-4, "Secure Hash Standard".	---	Ivanti MDM Android Client OpenSSL Component 2.2.1	#A6402
MDM Agent	HMAC-SHA2-384		---	Ivanti MDM Android Client OpenSSL Component 2.2.1	#A6402
MDM Server	HMAC-SHA2-256	Keyed-hash message authentication that meet the following: FIPS Pub 198-1, "The Keyed-Hash Message Authentication Code, and FIPS Pub 180-4, "Secure Hash Standard".	---	Ivanti MDM Bouncy Castle 1.0.2.4	#A6073
MDM Server	HMAC-SHA2-384		---	Ivanti MDM Bouncy Castle 1.0.2.4	#A6073

Platform Server or Agent	CAVP Algorithm	Standard	Modes Supported	Crypto Library	Cert #
MDM Server	HMAC-SHA2-256	Keyed-hash message authentication that meet the following: FIPS Pub 198-1, "The Keyed-Hash Message Authentication Code, and FIPS Pub 180-4, "Secure Hash Standard".	---	Ivanti MDM OpenSSL Component 1.1.1g	#A6074
MDM Server	HMAC-SHA2-384		---	Ivanti MDM OpenSSL Component 1.1.1g	#A6074
FCS_RBG_EXT.1/AGT Extended: Random Bit Generation *(MDM_PP)*					
MDM Agent	Counter DRBG	NIST Special Publication 800-90A	---	Ivanti MDM Android Client OpenSSL Component	#A6402
FCS_RBG_EXT.1/SRV Extended: Random Bit Generation *(MDM_PP)*					
MDM Server	Hash DRBG	NIST Special Publication 800-90A	---	Ivanti MDM Bouncy Castle 1.0.2.4	#A6073
MDM Server	Counter DRBG	NIST Special Publication 800-90A	---	Ivanti MDM OpenSSL Component 1.1.1g	#A6074

6.9 Distributed TOE SFR Allocation

For an MDM distributed TOE, the SFRs in the PP and Module must be met by the TOE as a whole. However, each TOE component will not necessarily meet each SFR. The following table specifies when each SFR must be implemented by a component.

The following categories are used to define the SFR allocations and defined in the cPP:

- All Components (All): All components that comprise of the distributed TOE must independently satisfy the requirement.
- At least one Component (One): This requirement must be fulfilled by at least one component within the distributed TOE.
- Feature Dependent (Feature Dependent): These requirements will only be fulfilled where the feature is implemented by the distributed TOE component.
- N/A: Indicated for iterated SFRs. Together the iterations indicate whether the requirement is met.

Table 29: Distributed TOE SFR Allocation

Requirement	SFR Allocation	MDM Server	MDM Agent
Security audit (FAU)			

Requirement	SFR Allocation	MDM Server	MDM Agent
FAU_ALT_EXT.1 Server Alerts *(MDM_PP)*	One	✓	
FAU_CRP_EXT.1 Support for Compliance Reporting of Mobile Device Configuration *(MDM_PP)*	One	✓	
FAU_GEN.1(1) Audit Data Generation *(MDM_PP)*	All	✓	✓
FAU_GEN.1(2)/SRV Audit Generation (MAS Server) *(MDM_PP)*	Feature Dependent	✓	
FAU_NET_EXT.1 Network Reachability Review *(MDM_PP)*	One	✓	
FAU_SAR.1 Audit Review *(MDM_PP)*	Feature Dependent	✓	
FAU_STG_EXT.1 External Trail Storage *(MDM_PP)*	All	✓	✓
FAU_STG_EXT.2 Audit Event Storage *(MDM_PP)*	Feature Dependent	✓	✓
Communication (FCO)			
FCO_CPC_EXT.1 Component Registration Channel Definition *(MDM_PP)*	All		
Cryptographic support (FCS)			
FCS_CKM.1 Cryptographic Key Generation *(MDM_PP)*	Feature Dependent	✓	✓
FCS_CKM.2 Cryptographic Key Establishment *(MDM_PP)*	All	✓	✓
FCS_CKM_EXT.4 Cryptographic Key Establishment *(MDM_PP)*	All	✓	✓
FCS_COP.1.1(1) Cryptographic Operation (Confidentiality Algorithms) *(MDM_PP)*	All	✓	✓
FCS_COP.1.1(2) Cryptographic Operation (Hashing Algorithms) *(MDM_PP)*	All	✓	✓
FCS_COP.1.1(3) Cryptographic Operation (Signature Algorithms) *(MDM_PP)*	All	✓	✓
FCS_COP.1.1(4) Cryptographic Operation (Keyed-Hash Message Authentication) *(MDM_PP)*	All	✓	✓
FCS_HTTPS_EXT.1 HTTPS Protocol`*(MDM_PP)*	Feature Dependent	✓	✓
FCS_IV_EXT.1 Initialization Vector Generation *(MDM_PP)*	Feature Dependent	✓	
FCS_RBG_EXT.1/AGT Extended: Random Bit Generation *(MDM_PP)*	All	N/A	✓
FCS_RBG_EXT.1/SRV Extended: Random Bit Generation *(MDM_PP)*	All	✓	N/A
FCS_STG_EXT.1 Cryptographic Key Storage *(MDM_PP)*	All	✓	✓

Requirement	SFR Allocation	MDM Server	MDM Agent
FCS_STG_EXT.2 Encrypted Cryptographic Key Storage *(MDM_PP)*	Feature Dependent	✓	
FCS_TLS_EXT.1 TLS Protocol *(TLS_PKG)*	Feature Dependent	✓	✓
FCS_TLSC_EXT.1 TLS Client Protocol *(TLS_PKG)*	Feature Dependent	✓	✓
FCS_TLSC_EXT.2 TLS Client Support for Mutual Authentication *(TLS_PKG)*	Feature Dependent	✓	✓
FCS_TLSC_EXT.5 TLS Client Support for Supported Groups Extension *(TLS_PKG)*	Feature Dependent	✓	✓
FCS_TLSS_EXT.1 TLS Server Protocol *(TLS_PKG)*	Feature Dependent	✓	
Identification and authentication (FIA)			
FIA_CLI_EXT.1 X.509 Unique Certificate *(MDM_PP)*	One		
FIA_ENR_EXT.1 Enrollment of Mobile Device into Management *(MDM_PP)*	One		
FIA_UAU.1 Timing of Authentication *(MDM_PP)*	One		
FIA_X509_EXT.1(1) X.509 Certification Validation *(MDM_PP)*	Feature Dependent	✓	✓
FIA_X509_EXT.2 X.509 Certificate Authentication *(MDM_PP)*	Feature Dependent	✓	✓
Security management (FMT)			
FMT_MOF.1(1) Management of Functions Behaviour *(MDM_PP)*	Feature Dependent	✓	
FMT_MOF.1(2) Management of functions Behaviour (Enrollment) *(MDM_PP)*	Feature Dependent	✓	
FMT_MOF.1(3) Management of Functions in (MAS Server Downloads) *(MDM_PP)*	Feature Dependent	✓	
FMT_POL_EXT.1 Trusted Policy Update *(MDM_PP)*	One	✓	
FMT_SMF.1(1) Specification of Management Functions (Server configuration of Agent) *(MDM_PP)*	One	✓	
FMT_SMF.1(2) Specification of Management Functions (Server Configuration of Server) *(MDM_PP)*	Feature Dependent	✓	
FMT_SMF.1(3) Specification of Management Functions (MAS Server) *(MDM_PP)*	Feature Dependent	✓	
FMT_SMR.1(1) Security Management Roles *(MDM_PP)*	One	✓	
FMT_SMR.1(2) Security Management Roles (MAS Server) *(MDM_PP)*	Feature Dependent	✓	

Requirement	SFR Allocation	MDM Server	MDM Agent
Protection of the TSF (FPT)			
FPT_API_EXT.1 Use of Supported Services and API's *(MDM_PP)*	All	✓	✓
FPT_ITT.1(2) Internal TOE TSF Data Transfer (MDM Agent) *(MDM_PP)*	Feature Dependent	✓	✓
FPT_LIB_EXT.1/AGT Use of Third Party Libraries *(MDM_PP)*	All	N/A	✓
FPT_LIB_EXT.1/SRV Use of Third Party Libraries *(MDM_PP)*	All	✓	N/A
FPT_TST_EXT.1 Functionality Testing *(MDM_PP)*	Feature Dependent Modified by TD0438	✓	
FPT_TUD_EXT.1 Trusted Update *(MDM_PP)*	All	✓	✓
Trusted path/channels (FTP)			
FTP_ITC.1(1) Inter-TSF Trusted Channel (Authorized IT Entities) *(MDM_PP)*	One	✓	
FTP_ITC_EXT.1 Trusted Channel *(MDM_PP)*	One	✓	
FTP_TRP.1(1) Trusted Path (for Remote Administration) *(MDM_PP)*	Feature Dependent	✓	
FTP_TRP.1(2) Trusted Path (for Enrollment) *(MDM_PP)*	Feature Dependent	✓	

7 Terms, Acronyms, and Abbreviations

Table 30: Terms

Term	Definition
Alert vs Event	An Alert is a asynchronous message sent verses an Event, which is a message sent as the result of an action.
Android Key Store	The Android Keystore system is a security feature that allows you to store cryptographic keys in a secure container, making them difficult to extract from the device. Once keys are stored in the keystore, they can be used for cryptographic operations while keeping the key material non-exportable.
Apps@Work	Apps@Work is an enterprise app storefront that facilitates the secure distribution of software and apps. It is integrated into Go app and Mobile@Work clients for iOS, Android and macOS.
Enterprise	The company data and applications the TOE protects.
MD User	A Mobile Device User.
MDM Agent	The name of the MDM Agent component of the TOE.
MDM System	The MDM Server and the MDM Agent.
Trusted Front End (TSF)	A Trusted Front End (TFE) is a TLS inspection proxy that is installed between the untrusted network and Ivanti EPMM. The TFE intercepts and decrypts HTTPS network traffic, and when it determines that the final destination is Ivanti EPMM, it re-encrypts and forwards the traffic to EPMM. The devices that register to Ivanti EPMM must send HTTPS requests to the TFE rather than to Ivanti EPMM. The devices must use port 443. Also, the TFE must be provisioned with digital certificates that establish an identity chain of trust with a legitimate server verified by a trusted third-party certificate authority.

Table 31: Acronyms and Abbreviations

Acronym	Definition
(AGT_PPM)	The SFR's source was MOD_MDM_AGENT_V1.0.
(MDM_PP)	The SFR's source is PP_MDM_V4.0.
(TLS_PKG)	The SFR's source is PKG_TLS_V1.1.
AES	Advanced Encryption Standard
API	Application Programming Interface
APK	Android Package
CAVP	Cryptographic Algorithm Validation Program
CC	Common Criteria
CN	Common Name
CRL	Certificate Revocation List
DN	Distinguished Name
DNS	Domain Name System

Acronym	Definition
DSS	Digital Signature Standard
ECC	Eliptic Curve Cryptography
ECDHE	Elliptic Curve Diffie-Hellman Ephemeral
ECDSA	Elliptic Curve Digital Signature Algorithm
EP	Extended Package
EPMM	Endpoint Mobile Manager
FIPS	Federal Information Processing Standard
FRSE	Freed or secure from harm
GCM	Galois/Counter Mode
GUI	Graphical User Interface
HMAC	Hash-based Message Authentication Code
HTTP	HyperText Transfer Protocol
HTTP/TLS	HTTP over TLS or HTTPS
HTTPS	HyperText Transfer Protocol Secure (HTTP over TLS)
IETF	Internet Engineering Task Force
IP	Internet Protocol
KAS-ECC-SCC	Key Agreement Scheme (KAS)
KeyGen	Key Generation
LAN	Local Area Network
MA	Mutual Authentication
MAM	Mobile Application Management
MAS	Mobile Application Store
MCM	Mobile Content Management
MD	Mobile Device
MDF	Mobile Device Fundamentals
MDM	Mobile Device Management
MIFS	MobileIron Admin Port (433 (default) or 8433)
MOD_MDM_AGENT_V1.0	*PP-Module for MDM Agents,* Version 1.0
NIAP	Nation Information Assurance Partnership
NIST	National Institute of Standards and Technology
N/A	Not Applicable
OCSP	Online Certificate Status Protocol
OE	Operational Environment

Acronym	Definition
OS	Operating System
PIN	Personal Identification Number
PKCS	Public-Key Cryptography Standard
PKG_TLS_V1.1	*Functional Package for Transport Layer Security (TLS),* Version 1.1
PP	Protection Profile
PP_MDM_V4.0	*Protection Profile for Mobile Device Management,* Version 4.0
PP-Module	Protection Profile Module
Pub	Publication
RBG	Random Bit Generation
RFC	Request for Comment (IETF)
RPM	RPM Package Manager
RSA	Rivest, Shamir & Adleman
SAN	Subject Alternative Name
SAR	Security Assurance Requirement
SHA	Secure Hash Algorithm
SigGen	Signature Generation
SigVer	Signature Verification
SFR	Security Functional Requirement
SMS	Short Message Server
SMTP	Simple Mail Trasfer Protocol
SP	Special Publication
SSH	Secure Shell
SSID	Service Set Identifier
ST	Security Target
TD	Technical Decisions
TLS	Transport Layer Security
TOE	Target of Evaluation
TSF	TOE Summary Functionality
TSS	TOE Summary Specification
URI	Uniform Resource Identifier
URL	Uniform Resource Locator
WiFi	Wireless Fidelity
WLAN	Wireless LAN

Acronym	Definition
WS	Work Station
XML	eXtensible Markup Language

Appendix A

This section provides more detailed information on the FAU_GEN.1(1) TSS and FAU_GEN.1(2)/AGT TSS Sections.

The MDM Server supports the following type of audit events.

- Administrative Actions: MDM Server
- Required Audit Events: MDM Server
- Administrative Actions: MAS Server
- Required Audit Events: MAS Server

The MDM Agents supports the following type of audit events.

- Required Audit Events: MDM Agent

The following tables identify the MDM Server audit events.

The following table identifies the administrative actions supported by the TOE. The audit event types are Administrative Actions:MDM Server and Administrative Actions:MAS Server. The MDM Server audits administrative functions that create, modify, or delete TSF data. Administrator functions that read TSF Data do not generate an audit record.

Table 32: Audit Event Type: Administrative Action: MDM Server

SFR	EPMM Server Mandatory Management Functions
FAU_NET_EXT.1 Network Reachability Review *(MDM_PP)*	Read the network connectivity status of an enrolled agent. AGD: Query Connectivity Status Audit: *Read TSF Data*
FAU_SAR.1 Audit Review *(MDM_PP)*	Enable authorized administrators the capability to read all audit data from the audit records and provide the audit records in a manner suitable for authorized administrators to interpret the information.
FAU_SEL.1(2) Security Audit Event Selection *(AGT_PPM)*	The TSF shall enable the ability for an administrator to select the set of events to be audited from the set of auditable events based on event type and success or failure of the auditable event.
FCO_CPC_EXT.1.1 Component Registration Channel Definition *(MDM_PP)*	The TSF shall implement functionality to require an Administrator to enable a secure communication channel between any pair of TOE component.
	The TSF shall implement functionality to implement a registration process in which components establish and use a communications channel.
FMT_MOF.1(3) Management of Functions in (MAS Server Downloads) *(MDM_PP)*	The MAS Server shall restrict the ability to download applications to enrolled mobile devices that are compliant with MDM policies and assigned to a user in the application access group to perform this function.
FMT_SMF.1.1(1) Specifications of Management Functions (Server configuration of Agent) *(MDM_PP)*	Transition to the locked state.
	Full Wipe of Protected Data.
	1. Unenroll the Android device.

SFR	EPMM Server Mandatory Management Functions
	2. Install policies on an Android Device.
	3. Query connectivity status of an Android device.
	4. Query the current version of the Android device's firmware/software.
	5. Query the current version of the Android device's hardware.
	6. Query the current version of the Android device's installed mobile applications.
	7. Import X.509v3 certificates into the Android device's Trust Anchor Database.
	8. Install an application on an Android device.
	9. Update the EPMM software on an Android device.
	10. Remove an application from an Android device.
	31. Update an Android device's Password Policy that includes specifying: • Minimum password length • Minimum password complexity and • Maximum password lifetime.
	32. Update an Android device's Session Locking Policy that includes specifying: • Screen-lock enable/disable, • Screen-lock timeout, • Number of authentication attempts.
	33. Update an Android device's wireless network Policy that defines which networks an Android device may join.
	34. Update the Security Policy for each wireless network that a. specifies the FQDN of acceptable WLAN authentication server certificates.

SFR	EPMM Server Mandatory Management Functions
	b. Specifies the security type. c. Specifies the authentication protocol. d. Specifies the client credentials to be used for authentication.
	35. Specify an application installation Policy (MAS) that specifies an authorized application repository(s).
	36. Update an Android device's enable/disable Policy for • Camera • Microphone
FMT_SMF.1.1(2) Specifications of Management Functions (Server configuration of Server) *(MDM_PP)*	b. Configure devices by specifying o a unique device ID o a number of devices
FMT_SMF.1.1(3) Specifications of Management Functions (MAS Server) *(MDM_PP)*	a) Configure application access groups
	b) Download applications.
FMT_SMF_EXT.4.1 Specification of Management Functions *(AGT_PPM)*	Import the certificates to be used for authentication of MDM Agent communications
FMT_SMF_EXT.4.2 Specifications of Management Functions (MAS Server) *(MDM_PP)*	Enroll in management
	Configure whether users can unenroll from management
	Configure periodicity of reachability events.
FMT_TUD_EXT.1.1 Trusted Update *(MDM_PP)*	The TSF shall provide Authorized Administrators with the ability to query the current version of software.
	The TSF shall implement functionality to provide Authorized Administrators with the ability to initiate updates to TSF software.

The following table identifies the Required Audit Events: MDM Server event audit records generated by the MDM Server. Refer to AGD Section 11 EPMM System Audit Record Examples Table 10 Audible events for MDM server for examples of the audit records.

Table 33: MDM Server Security Functional Requirements and Auditable Events

Requirement	Auditable Events	Additional Audit Record Contents
Security audit (FAU)		
FAU_ALT_EXT.1	Type of alert.	Identity of Mobile Device that sent
Communication (FCO)		
FCO_CPC_EXT.1	Enabling or disabling communications between a pair of components	Identities of the endpoints pairs enabled or disabled.
Cryptographic Support (FCS)		
FCS_HTTPS_EXT.1	Failure of the certificate validity check.	Issuer Name and Subject Name of certificate. [no additional information]
FCS_RBG_EXT.1/SRV	Failure of the randomization process.	No additional information.
FCS_TLSC_EXT.1	Failure to establish a TLS session.	Reason for failure.
	Failure to verify presented identifier.	Presented identifier and reference identifier.
FCS_TLSS_EXT.1	Failure to establish a TLS session.	Reason for failure.
Identification and authentication (FIA)		
FIA_ENR_EXT.1	Failure of MD user authentication.	Presented username.
FIA_X509_EXT.1(1)	Failure to validate X.509 certificate.	Reason for failure.
FIA_X509_EXT.2	Failure to establish connection to determine revocation status.	No additional information.
Security management (FMT)		
FMT_MOF.1(1)	Issuance of command to perform function.	Command sent and identity of MDM Agent recipient(s).
	Change of policy settings.	Policy changed and value or full policy.
FMT_MOF.1(2)	Enrollment by a user.	Identity of user
FMT_SMF.1(2)	Success or failure of function.	No additional information.
Protection of the TSF (FPT)		
FPT_ITT.1(2)	Initiation and termination of the trusted channel.	Trusted channel protocol. Identity of initiator and recipient.
FPT_TST_EXT.1	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	Success or failure of signature verification.	No additional information
Trusted path/channels (FTP)		
FTP_ITC.1(1)	Initiation and termination of the trusted channel.	Trusted channel protocol. Non-TOE endpoint of connection.

Requirement	Auditable Events	Additional Audit Record Contents
FTP_TRP.1(1)	Initiation and termination of the trusted channel.	Trusted channel protocol. Identity of administrator.
FTP_TRP.1(2)	Initiation and termination of the trusted channel.	Trusted channel protocol.

Table 34: Events and messages

Event	Description of when the event is logged	Severity
Audit configuration change	A change in audit configuration (which is part of Samsung General Policy)	Alert
Transition to locked state	Device becomes locked either due to user action on the device or administrator action on EPMM.	Transition to locked state
Policy validation failure	Finding a discrepancy in policy validation.	Error for policy errors. Warning for policy warnings.
CA certificate import	Installation (success or failure) of CA certificates.	Notice if success. Error if failure.
ID certificate import	Installation (success or failure) of identity certificates.	Notice if success. Error if. failure
CA certificate remove	Removal (success or failure) of CA certificates.	Notice if success. Error if failure.
ID certificate remove	Removal (success or failure) of identity certificates.	Notice
Install application	Installation of an application.	Notice
Remove application	Removal of an application.	Notice
Device OS upgrade	Device system software upgrade.	Notice
Server request client checkin	Logs wakeup command from EPMM that results in a device checkin from Federal build.	Notice
Request current software version	Attempt to fetch and report current software version.	Notice
Request current hardware version	Attempt to fetch and report current hardware version.	Notice
Report app inventory	Attempt to report apps inventory.	Notice if success. Error if failure
Upload logs	Attempt to upload logs to server.	Notice if success. Error if failure.

Event	Description of when the event is logged	Severity
Configure App Store quarantine compliance action	Apps@Work (in Federal build) is quarantined or taken out of quarantine.	Notice
Enable/Disable camera	Enabling/disabling camera.	Notice
Enable/Disable mic	Enabling/disabling microphone.	Notice
Enable/Disable radios	Enabling/disabling any of Wi-Fi, NFC, cellular, or Bluetooth.	Notice
Enable/Disable development mode	Enabling/disabling development mode.	Notice
Configure unlock banner	Changing and applying unlock banner policy.	Notice if success. Error if failure.
Enable/Disable usb tether	Enabling/disabling of USB tethering.	Notice
Enable/Disable location services	Enabling/disabling of location services.	Notice
Enable/Disable biometrics	Enabling/disabling biometric authentication.	Notice if success. Error if failure.
Failure to establish a TLS session	Failure in establishing TLS session, including noting any certificate error.	Warning
TLS session established	Established TLS session with EPMM.	Notice
TLS session terminated	Terminated TLS session with EPMM	Warning
Failure to verify presented identity	Failure to verify presented identity	Warning
Android shutdown	Android is shutting down. This event belongs to the System audit group and is logged if the System group is selected in the **Audit Groups** field of the Samsung General Policy These events are logged by the Samsung device rather than Federal build so the format is different.	Notice
Android startup completed	Android startup has completed. This event belongs to the System audit group and is logged if the System group is selected in the **Audit Groups** field of the Samsung General Policy These events are logged by the Samsung device rather than Federal build so the format is different.	Notice

Table 35: EPMM Agent *(AGT_PPM)* Security Functional Requirements and Auditable Events (addressed TD0660)

Requirement	SFR Source PP/Module /Functional Package	Auditable Events	Additional Audit Record Contents
Security Audit (FAU)			
FAU_ALT_EXT.2	MOD_MDM_AGENT_V1.0	Success/failure of sending alert.	No additional information.
FAU_SEL.1(2)	MOD_MDM_AGENT_V1.0	All modifications to the audit configuration that occur while the audit collection functions are operating.	No additional information.
Cryptographic Support (FCS)			
FCS_HTTPS_EXT.1	PP_MDM_V4.0	Failure of the certificate validity check.	Issuer Name and Subject Name of certificate. [no additional information]
FCS_RBG_EXT.1/AGT	PP_MDM_V4.0	Failure of the randomization process.	No additional information.
FCS_TLSC_EXT.1	PKG_TLS_v1.1	Failure to establish a TLS session.	Reason for failure.
		Failure to verify presented identifier.	Presented identifier and reference identifier.
		Establishment/termination of a TLS session	Non-TOE endpoint of connection.
Identification and authentication (FIA)			
FIA_ENR_EXT.2	MOD_MDM_AGENT_V1.0	Enrollment in management.	Reference identifier of MDM Server.
FIA_X509_EXT.1(1)	PP_MDM_V4.0	Failure to validate X.509 certificate.	Reason for failure.

Requirement	SFR Source PP/Module /Functional Package	Auditable Events	Additional Audit Record Contents
FIA_X509_EXT.2	PP_MDM_V4.0	Failure to establish connection to determine revocation status.	No additional information.
Security management (FMT)			
FMT_POL_EXT.2	MOD_MDM_AGENT_V1.0	Failure of policy validation.	Reason for failure of validation.
FMT_SMF_EXT.4	MOD_MDM_AGENT_V1.0	Outcome (Success/failure) of function.	No additional information.
FMT_UNR_EXT.1	MOD_MDM_AGENT_V1.0	[Attempt to unenroll]	No additional information.
Protection of the TSF (FPT)			
FPT_TUD_EXT.1	PP_MDM_V4.0	Success or failure of signature verification.	No additional information

Appendix B

This Appendix lists the third-party libraries that are included in the two TOE component builds.

android-market-api:android-market-api:jar:0.6
antlr:antlr:jar:2.7.7
axis:axis-wsdl4j:jar:1.5.1
axis:axis:jar:1.4
cglib:cglib:jar:3.3.0
ch.qos.logback:logback-classic:jar:1.2.9
ch.qos.logback:logback-core:jar:1.2.9
co.elastic.clients:elasticsearch-java:jar:7.17.3
com.amazonaws:aws-java-sdk-cloudfront:jar:1.12.506
com.amazonaws:aws-java-sdk-core:jar:1.12.506
com.amazonaws:aws-java-sdk-kms:jar:1.12.506
com.amazonaws:aws-java-sdk-route53:jar:1.12.506
com.amazonaws:aws-java-sdk-s3:jar:1.12.506
com.amazonaws:jmespath-java:jar:1.12.506
com.android.tools.build:apksig:jar:7.2.2
com.azure:azure-core-http-netty:jar:1.13.11
com.azure:azure-core:jar:1.45.1
com.azure:azure-json:jar:1.1.0
com.azure:azure-storage-blob:jar:12.25.1
com.azure:azure-storage-common:jar:12.24.1
com.azure:azure-storage-internal-avro:jar:12.10.1
com.beust:jcommander:jar:1.48
com.carrotsearch:hppc:jar:0.8.1
com.caucho:hessian:jar:4.0.63
com.cronutils:cron-utils:jar:9.1.6
com.fasterxml.jackson.core:jackson-annotations:jar:2.14.1
com.fasterxml.jackson.core:jackson-annotations:jar:2.16.1
com.fasterxml.jackson.core:jackson-core:jar:2.14.1
com.fasterxml.jackson.core:jackson-core:jar:2.16.1
com.fasterxml.jackson.core:jackson-databind:jar:2.14.1
com.fasterxml.jackson.core:jackson-databind:jar:2.16.1
com.fasterxml.jackson.dataformat:jackson-dataformat-cbor:jar:2.10.4
com.fasterxml.jackson.dataformat:jackson-dataformat-cbor:jar:2.12.6
com.fasterxml.jackson.dataformat:jackson-dataformat-smile:jar:2.10.4
com.fasterxml.jackson.dataformat:jackson-dataformat-smile:jar:2.9.8

com.fasterxml.jackson.dataformat:jackson-dataformat-yaml:jar:2.10.4
com.fasterxml.jackson.datatype:jackson-datatype-jdk8:jar:2.14.1
com.fasterxml.jackson.datatype:jackson-datatype-jdk8:jar:2.16.1
com.fasterxml.jackson.datatype:jackson-datatype-jsr310:jar:2.16.1
com.fasterxml.jackson.datatype:jackson-datatype-jsr310:jar:2.9.8
com.fasterxml.jackson.jaxrs:jackson-jaxrs-base:jar:2.9.8
com.fasterxml.jackson.jaxrs:jackson-jaxrs-json-provider:jar:2.9.8
com.fasterxml.jackson.module:jackson-module-jaxb-annotations:jar:2.9.8
com.fasterxml.jackson.module:jackson-module-mrbean:jar:2.9.8
com.fasterxml.jackson.module:jackson-module-parameter-names:jar:2.16.1
com.fasterxml.woodstox:woodstox-core:jar:5.0.3
com.fasterxml:classmate:jar:1.4.0
com.fasterxml:classmate:jar:1.5.1
com.github.ben-manes.caffeine:caffeine:jar:2.7.0
com.github.stephenc.jcip:jcip-annotations:jar:1.0-1
com.github.virtuald:curvesapi:jar:1.06
com.google.android:annotations:jar:4.1.1.4
com.google.api-client:google-api-client:jar:1.35.0
com.google.api.grpc:proto-google-cloud-pubsub-v1:jar:1.105.12
com.google.api.grpc:proto-google-common-protos:jar:2.18.0
com.google.api.grpc:proto-google-iam-v1:jar:1.13.0
com.google.api:api-common:jar:2.10.0
com.google.api:gax-grpc:jar:2.27.0
com.google.api:gax-httpjson:jar:0.112.0
com.google.api:gax:jar:2.27.0
com.google.apis:google-api-services-admin-directory:jar:directory_v1-rev20221108-2.0.0
com.google.apis:google-api-services-androidenterprise:jar:v1-rev20230111-2.0.0
com.google.apis:google-api-services-androidmanagement:jar:v1-rev20230529-2.0.0
com.google.auth:google-auth-library-credentials:jar:0.22.2
com.google.auth:google-auth-library-oauth2-http:jar:0.22.2
com.google.auto.value:auto-value-annotations:jar:1.10.1
com.google.cloud:google-cloud-pubsub:jar:1.123.12
com.google.code.findbugs:annotations:jar:3.0.1
com.google.code.findbugs:jsr305:jar:3.0.2
com.google.code.gson:gson:jar:1.6
com.google.errorprone:error_prone_annotations:jar:2.18.0
com.google.errorprone:error_prone_annotations:jar:2.5.1
com.google.guava:failureaccess:jar:1.0.1
com.google.guava:guava:jar:30.1.1-jre

com.google.guava:guava:jar:32.1.1-jre
com.google.guava:listenablefuture:jar:9999.0-empty-to-avoid-conflict-with-guava
com.google.http-client:google-http-client-apache-v2:jar:1.41.8
com.google.http-client:google-http-client-gson:jar:1.41.8
com.google.http-client:google-http-client-jackson2:jar:1.38.1
com.google.http-client:google-http-client:jar:1.41.8
com.google.j2objc:j2objc-annotations:jar:1.3
com.google.j2objc:j2objc-annotations:jar:2.8
com.google.oauth-client:google-oauth-client:jar:1.33.3
com.google.protobuf:protobuf-java-util:jar:3.21.12
com.google.protobuf:protobuf-java:jar:3.21.12
com.google.re2j:re2j:jar:1.6
com.googlecode.concurrentlinkedhashmap:concurrentlinkedhashmap-lru:jar:1.4.2
com.googlecode.javaewah:JavaEWAH:jar:1.1.6
com.googlecode.json-simple:json-simple:jar:1.1.1
com.googlecode.juniversalchardet:juniversalchardet:jar:1.0.3
com.googlecode.libphonenumber:libphonenumber:jar:8.12.8
com.googlecode.plist:dd-plist:jar:1.16
com.googlecode.protobuf-java-format:protobuf-java-format:jar:1.4
com.jayway.jsonpath:json-path:jar:2.6.0
com.jayway.jsonpath:json-path:jar:2.9.0
com.jcraft:jsch:jar:0.1.54
com.jolbox:bonecp:jar:0.7.1.RELEASE
com.mchange:c3p0:jar:0.9.5.5
com.mchange:mchange-commons-java:jar:0.3.0
com.mi.platform:common-vsp:jar:12.3.1.0-SNAPSHOT
com.mi.platform:common:jar:12.3.1.0-SNAPSHOT
com.mi.platform:entrust:jar:11.0.0
com.mi.platform:integration-ca-resources:jar:1.0.0.0-SNAPSHOT
com.mi.platform:integration-ca:jar:96.0.0-SNAPSHOT
com.mi.platform:mics-core:jar:12.3.1.0-SNAPSHOT
com.mi.platform:test-shared:jar:12.3.1.0-SNAPSHOT
com.mobileiron.bouncycastle:bcmail-fips:jar:1.0.3
com.mobileiron.bouncycastle:bcpkix-fips:jar:1.0.3
com.mobileiron.bouncycastle:bctls-fips:jar:1.0.9
com.mobileiron.external:jscep:jar:2.4.2.1-SNAPSHOT
com.mobileiron.external:jscep:jar:2.4.3.1-SNAPSHOT
com.mobileiron.external:pecoff4j:jar:0.0.2.1
com.mobileiron.mirror.apktool:apktool-lib:jar:2.6.0

com.mobileiron.mirror.apktool:brut.j.common:jar:2.6.0
com.mobileiron.mirror.apktool:brut.j.dir:jar:2.6.0
com.mobileiron.mirror.apktool:brut.j.util:jar:2.6.0
com.mobileiron.springframework.security.extensions:spring-security-saml2-core:jar:1.1.0.BUILD-SNAPSHOT
com.mobileiron.windows:windows-service-lib:jar:1.0.10-SNAPSHOT
com.mobileiron:code-quality:jar:1.0.28-SNAPSHOT
com.mobileiron:common-swagger:jar:103.0.16
com.mobileiron:component-access-integration:jar:103.0.21
com.mobileiron:component-accessory:jar:103.0.21
com.mobileiron:component-base:jar:103.0.16
com.mobileiron:component-common:jar:103.0.21
com.mobileiron:component-contact:jar:103.0.21
com.mobileiron:component-contract:jar:78.0.21
com.mobileiron:component-gdpr:jar:103.0.21
com.mobileiron:component-gpo:jar:103.0.21
com.mobileiron:component-intune-compliance:jar:88.0.6
com.mobileiron:component-mtd:jar:103.0.21
com.mobileiron:component-office365-dlp:jar:98.0.0
com.mobileiron:component-script:jar:103.0.21
com.mobileiron:component-software-update:jar:101.0.0
com.mobileiron:jaxb-adapters:jar:103.0.0
com.mobileiron:polaris-common-activemq-plugin:jar:108.0.6
com.mobileiron:polaris-common-apk:jar:103.0.22
com.mobileiron:polaris-common-apple-dep:jar:103.0.22
com.mobileiron:polaris-common-apple-vpp:jar:103.0.22
com.mobileiron:polaris-common-base:jar:103.0.22
com.mobileiron:polaris-common-base:jar:108.0.6
com.mobileiron:polaris-common-cloud-ca:jar:95.0.13
com.mobileiron:polaris-common-crypto:jar:103.0.22
com.mobileiron:polaris-common-crypto:jar:108.0.6
com.mobileiron:polaris-common-google-androidwork:jar:103.0.22
com.mobileiron:polaris-common-google-play-search:jar:103.0.22
com.mobileiron:polaris-common-google:jar:100.0.4
com.mobileiron:polaris-common-google:jar:103.0.22
com.mobileiron:polaris-common-hydra:jar:103.0.16
com.mobileiron:polaris-common-jackson:jar:103.0.22
com.mobileiron:polaris-common-jackson:jar:108.0.6
com.mobileiron:polaris-common-jms-activemq:jar:103.0.16

com.mobileiron:polaris-common-jms:jar:103.0.16
com.mobileiron:polaris-common-jwt:jar:103.0.22
com.mobileiron:polaris-common-logging:jar:103.0.22
com.mobileiron:polaris-common-managed-app-config:jar:103.0.22
com.mobileiron:polaris-common-microsoft-bsp:jar:103.0.22
com.mobileiron:polaris-common-microsoft-graph:jar:97.0.5
com.mobileiron:polaris-common-net:jar:100.0.4
com.mobileiron:polaris-common-net:jar:103.0.22
com.mobileiron:polaris-common-net:jar:108.0.6
com.mobileiron:polaris-common-pkg:jar:103.0.22
com.mobileiron:polaris-common-plist:jar:103.0.22
com.mobileiron:polaris-common-policy:jar:103.0.22
com.mobileiron:polaris-common-rest-support:jar:103.0.22
com.mobileiron:polaris-common-samsung-efota:jar:103.0.22
com.mobileiron:polaris-common-services:jar:100.0.4
com.mobileiron:polaris-common-services:jar:103.0.22
com.mobileiron:polaris-common-spring-http:jar:103.0.22
com.mobileiron:polaris-common-spring-http:jar:108.0.6
com.mobileiron:polaris-common-spring-ws:jar:103.0.22
com.mobileiron:polaris-common-teamviewer:jar:103.0.22
com.mobileiron:polaris-common-tenant-suppress:jar:100.0.4
com.mobileiron:polaris-common-tenant-suppress:jar:108.0.6
com.mobileiron:polaris-common-tenant-suppress:jar:97.0.25
com.mobileiron:polaris-common-windows-apps:jar:103.0.22
com.mobileiron:polaris-common-windows-xml:jar:103.0.22
com.mobileiron:polaris-common-zdt-liquibase:jar:103.0.22
com.mobileiron:polaris-common-zebra-fota:jar:103.0.22
com.mobileiron:polaris-common:jar:103.0.22
com.mobileiron:protocol-afwcore:jar:2.0.0-SNAPSHOT
com.mobileiron:protocol-androidclient:jar:103.0.3
com.mobileiron:protocol-appconnect:jar:1.0.14a-SNAPSHOT
com.mobileiron:protocol-appconnect:jar:1.0.17-SNAPSHOT
com.mobileiron:protocol:jar:95.0.4
com.nimbusds:nimbus-jose-jwt:jar:9.37.3
com.nulab-inc:zxcvbn:jar:1.2.2
com.opencsv:opencsv:jar:3.4
com.ryantenney.metrics:metrics-spring:jar:3.1.3
com.splunk:splunk:jar:1.6.5.0
com.squareup.okhttp3:okhttp:jar:4.11.0

com.squareup.okio:okio-jvm:jar:3.2.0
com.squareup.okio:okio:jar:3.2.0
com.squareup.okio:okio:jar:3.4.0
com.sun.activation:jakarta.activation:jar:1.2.2
com.sun.istack:istack-commons-runtime:jar:3.0.7
com.sun.istack:istack-commons-runtime:jar:3.0.8
com.sun.mail:javax.mail:jar:1.5.4
com.sun.xml.bind:jaxb-xjc:jar:2.2.10
com.sun.xml.fastinfoset:FastInfoset:jar:1.2.15
com.sun.xml.fastinfoset:FastInfoset:jar:1.2.16
com.sun.xml.messaging.saaj:saaj-impl:jar:1.5.3
com.sun:ldapbp:jar:1.0
com.tdunning:t-digest:jar:3.2
com.thoughtworks.qdox:qdox:jar:1.6.3
com.thoughtworks.xstream:xstream:jar:1.4.20
com.twelvemonkeys.common:common-image:jar:3.7.1
com.twelvemonkeys.common:common-io:jar:3.7.1
com.twelvemonkeys.common:common-lang:jar:3.7.1
com.twelvemonkeys.imageio:imageio-bmp:jar:3.7.1
com.twelvemonkeys.imageio:imageio-core:jar:3.7.1
com.twelvemonkeys.imageio:imageio-icns:jar:3.7.1
com.twelvemonkeys.imageio:imageio-metadata:jar:3.7.1
com.twelvemonkeys.imageio:imageio-tiff:jar:3.7.1
com.zaxxer:HikariCP-java7:jar:2.4.13
com.zaxxer:SparseBitSet:jar:1.2
commons-beanutils:commons-beanutils:jar:1.9.4
commons-chain:commons-chain:jar:1.2
commons-cli:commons-cli:jar:1.2
commons-codec:commons-codec:jar:1.15
commons-collections:commons-collections:jar:3.2.2
commons-configuration:commons-configuration:jar:1.7
commons-dbcp:commons-dbcp:jar:1.4
commons-digester:commons-digester:jar:1.8
commons-digester:commons-digester:jar:1.8.1
commons-discovery:commons-discovery:jar:0.2
commons-fileupload:commons-fileupload:jar:1.5
commons-io:commons-io:jar:2.7
commons-io:commons-io:jar:2.8.0
commons-lang:commons-lang:jar:2.4

commons-lang:commons-lang:jar:2.6
commons-logging:commons-logging:jar:1.1.3
commons-net:commons-net:jar:20030805.205232
commons-net:commons-net:jar:3.0
commons-pool:commons-pool:jar:1.5.4
commons-validator:commons-validator:jar:1.5.1
commons-validator:commons-validator:jar:1.7
io.dropwizard.metrics:metrics-annotation:jar:3.1.2
io.dropwizard.metrics:metrics-core:jar:3.1.2
io.dropwizard.metrics:metrics-core:jar:3.2.5
io.dropwizard.metrics:metrics-healthchecks:jar:3.1.2
io.github.x-stream:mxparser:jar:1.2.2
io.grpc:grpc-alts:jar:1.54.0
io.grpc:grpc-api:jar:1.54.0
io.grpc:grpc-auth:jar:1.54.0
io.grpc:grpc-context:jar:1.27.2
io.grpc:grpc-context:jar:1.54.0
io.grpc:grpc-core:jar:1.54.0
io.grpc:grpc-googleapis:jar:1.54.0
io.grpc:grpc-grpclb:jar:1.54.0
io.grpc:grpc-netty-shaded:jar:1.54.0
io.grpc:grpc-protobuf-lite:jar:1.54.0
io.grpc:grpc-protobuf:jar:1.54.0
io.grpc:grpc-services:jar:1.54.0
io.grpc:grpc-stub:jar:1.39.0
io.grpc:grpc-xds:jar:1.54.0
io.jsonwebtoken:jjwt-api:jar:0.12.2
io.jsonwebtoken:jjwt-impl:jar:0.12.2
io.jsonwebtoken:jjwt-jackson:jar:0.12.2
io.jsonwebtoken:jjwt:jar:0.12.2
io.netty:netty-buffer:jar:4.1.100.Final
io.netty:netty-codec-dns:jar:4.1.101.Final
io.netty:netty-codec-http2:jar:4.1.100.Final
io.netty:netty-codec-http2:jar:4.1.101.Final
io.netty:netty-codec-http:jar:4.1.100.Final
io.netty:netty-codec-http:jar:4.1.101.Final
io.netty:netty-codec-socks:jar:4.1.100.Final
io.netty:netty-codec-socks:jar:4.1.101.Final
io.netty:netty-codec:jar:4.1.100.Final

io.netty:netty-common:jar:4.1.100.Final
io.netty:netty-handler-proxy:jar:4.1.100.Final
io.netty:netty-handler-proxy:jar:4.1.101.Final
io.netty:netty-handler:jar:4.1.100.Final
io.netty:netty-resolver-dns-classes-macos:jar:4.1.101.Final
io.netty:netty-resolver-dns-native-macos:jar:osx-x86_64
io.netty:netty-resolver-dns:jar:4.1.101.Final
io.netty:netty-resolver:jar:4.1.100.Final
io.netty:netty-tcnative-boringssl-static:jar:2.0.62.Final
io.netty:netty-tcnative-boringssl-static:jar:linux-aarch_64
io.netty:netty-tcnative-boringssl-static:jar:linux-x86_64
io.netty:netty-tcnative-boringssl-static:jar:osx-aarch_64
io.netty:netty-tcnative-boringssl-static:jar:osx-x86_64
io.netty:netty-tcnative-boringssl-static:jar:windows-x86_64
io.netty:netty-tcnative-classes:jar:2.0.62.Final
io.netty:netty-transport-classes-epoll:jar:4.1.101.Final
io.netty:netty-transport-classes-kqueue:jar:4.1.101.Final
io.netty:netty-transport-native-epoll:jar:linux-x86_64
io.netty:netty-transport-native-kqueue:jar:osx-x86_64
io.netty:netty-transport-native-unix-common:jar:4.1.100.Final
io.netty:netty-transport:jar:4.1.100.Final
io.opencensus:opencensus-api:jar:0.31.0
io.opencensus:opencensus-api:jar:0.31.1
io.opencensus:opencensus-contrib-http-util:jar:0.31.0
io.opencensus:opencensus-contrib-http-util:jar:0.31.1
io.opencensus:opencensus-proto:jar:0.2.0
io.perfmark:perfmark-api:jar:0.26.0
io.projectreactor.netty:reactor-netty-core:jar:1.0.39
io.projectreactor.netty:reactor-netty-http:jar:1.0.39
io.projectreactor:reactor-core:jar:3.4.34
io.springfox:springfox-core:jar:2.9.2
io.springfox:springfox-schema:jar:2.9.2
io.springfox:springfox-spi:jar:2.9.2
io.springfox:springfox-spring-web:jar:2.9.2
io.springfox:springfox-swagger-common:jar:2.9.2
io.springfox:springfox-swagger-ui:jar:2.9.2
io.springfox:springfox-swagger2:jar:2.9.2
io.swagger:swagger-annotations:jar:1.5.20
io.swagger:swagger-models:jar:1.5.20

io.zipkin.brave:brave-instrumentation-http:jar:5.6.0
io.zipkin.brave:brave-instrumentation-httpclient:jar:5.6.0
io.zipkin.brave:brave:jar:5.6.0
io.zipkin.reporter2:zipkin-reporter:jar:2.7.13
io.zipkin.zipkin2:zipkin:jar:2.11.10
jakarta.activation:jakarta.activation-api:jar:1.2.1
jakarta.annotation:jakarta.annotation-api:jar:1.3.5
jakarta.json:jakarta.json-api:jar:2.0.1
jakarta.jws:jakarta.jws-api:jar:2.1.0
jakarta.ws.rs:jakarta.ws.rs-api:jar:2.1.6
jakarta.xml.bind:jakarta.xml.bind-api:jar:2.3.2
jakarta.xml.soap:jakarta.xml.soap-api:jar:1.4.2
jakarta.xml.ws:jakarta.xml.ws-api:jar:2.3.3
javax.activation:activation:jar:1.1
javax.activation:javax.activation-api:jar:1.2.0
javax.annotation:javax.annotation-api:jar:1.3.2
javax.annotation:jsr250-api:jar:1.0
javax.ejb:ejb-api:jar:3.0-alpha-1
javax.inject:javax.inject:jar:1
javax.jms:javax.jms-api:jar:2.0.1
javax.json:javax.json-api:jar:1.0
javax.management.j2ee:management-api:jar:1.1-rev-1
javax.persistence:javax.persistence-api:jar:2.2
javax.persistence:persistence-api:jar:1.0
javax.servlet:javax.servlet-api:jar:3.1.0
javax.servlet:jstl:jar:1.2
javax.validation:validation-api:jar:2.0.1.Final
javax.ws.rs:jsr311-api:jar:1.1
javax.xml.bind:jaxb-api:jar:2.3.0
javax.xml.bind:jaxb-api:jar:2.3.1
javax.xml.bind:jsr173_api:jar:1.0
javax.xml.stream:stax-api:jar:1.0-2
javax.xml.ws:jaxws-api:jar:2.0
javax.xml:jaxrpc-api:jar:1.1
jaxen:jaxen:jar:1.1.1
jdom:jdom:jar:1.0
jline:jline:jar:2.12
jmock:jmock:jar:1.1.0
joda-time:joda-time:jar:2.10.3

joda-time:joda-time:jar:2.8.1
junit:junit:jar:3.8.1
junit:junit:jar:4.10
net.bytebuddy:byte-buddy-agent:jar:1.14.5
net.bytebuddy:byte-buddy:jar:1.10.17
net.bytebuddy:byte-buddy:jar:1.14.5
net.coobird:thumbnailator:jar:0.4.8
net.grey-panther:natural-comparator:jar:1.1
net.java.dev.jna:jna:jar:5.10.0
net.jcip:jcip-annotations:jar:1.0
net.minidev:accessors-smart:jar:2.4.7
net.minidev:json-smart:jar:2.4.11
net.minidev:json-smart:jar:2.4.7
net.sf.ehcache:ehcache:jar:2.10.6
net.sf.ezmorph:ezmorph:jar:1.0.4
net.sf.jopt-simple:jopt-simple:jar:5.0.2
net.sf.json-lib:json-lib-ext-spring:jar:1.0.2
net.sf.json-lib:json-lib:jar:jdk15
net.sf.plist:propertylist:jar:2.0.0
net.sf.saxon:saxon-dom:jar:9.0
net.sf.saxon:saxon:jar:9
net.sf.supercsv:super-csv:jar:2.4.0
net.shibboleth.ext:spring-extensions:jar:5.4.2
net.shibboleth.utilities:java-support:jar:7.3.0
net.sourceforge.findbugs:annotations:jar:1.3.2
net.spy:spymemcached:jar:2.11.4
nl.jqno.equalsverifier:equalsverifier:jar:3.14.3
org.antlr:antlr-runtime:jar:3.5
org.antlr:antlr4-runtime:jar:4.7
org.antlr:stringtemplate:jar:3.2.1
org.apache.abdera:abdera-client:jar:1.1.2
org.apache.abdera:abdera-core:jar:1.1.2
org.apache.abdera:abdera-i18n:jar:1.1.2
org.apache.abdera:abdera-parser:jar:1.1.2
org.apache.activemq:activemq-broker:jar:5.16.7
org.apache.activemq:activemq-client:jar:5.16.7
org.apache.activemq:activemq-jms-pool:jar:5.16.7
org.apache.activemq:activemq-openwire-legacy:jar:5.16.7
org.apache.activemq:activemq-pool:jar:5.16.7

org.apache.activemq:activemq-spring:jar:5.16.7
org.apache.ant:ant-launcher:jar:1.10.14
org.apache.ant:ant:jar:1.10.14
org.apache.axis:axis-jaxrpc:jar:1.4
org.apache.axis:axis-saaj:jar:1.4
org.apache.axis:axis:jar:1.4
org.apache.commons:commons-collections4:jar:4.4
org.apache.commons:commons-compress:jar:1.19
org.apache.commons:commons-exec:jar:1.0
org.apache.commons:commons-lang3:jar:3.12.0
org.apache.commons:commons-lang3:jar:3.6
org.apache.commons:commons-math3:jar:3.6.1
org.apache.commons:commons-pool2:jar:2.6.2
org.apache.commons:commons-text:jar:1.10.0
org.apache.cxf:cxf-core:jar:3.5.5
org.apache.cxf:cxf-rt-bindings-soap:jar:3.5.9
org.apache.cxf:cxf-rt-bindings-xml:jar:3.5.9
org.apache.cxf:cxf-rt-databinding-jaxb:jar:3.5.5
org.apache.cxf:cxf-rt-frontend-jaxrs:jar:3.5.9
org.apache.cxf:cxf-rt-frontend-jaxws:jar:3.5.9
org.apache.cxf:cxf-rt-frontend-simple:jar:3.5.9
org.apache.cxf:cxf-rt-rs-extension-providers:jar:3.5.9
org.apache.cxf:cxf-rt-security:jar:3.5.9
org.apache.cxf:cxf-rt-transports-http:jar:3.5.9
org.apache.cxf:cxf-rt-ws-addr:jar:3.5.9
org.apache.cxf:cxf-rt-ws-policy:jar:3.5.9
org.apache.cxf:cxf-rt-wsdl:jar:3.5.9
org.apache.geronimo.specs:geronimo-activation_1.0.2_spec:jar:1.1
org.apache.geronimo.specs:geronimo-activation_1.1_spec:jar:1.0.2
org.apache.geronimo.specs:geronimo-j2ee-management_1.1_spec:jar:1.0.1
org.apache.geronimo.specs:geronimo-jms_1.1_spec:jar:1.1.1
org.apache.geronimo.specs:geronimo-jta_1.1_spec:jar:1.1.1
org.apache.geronimo.specs:geronimo-stax-api_1.0_spec:jar:1.0.1
org.apache.httpcomponents:httpasyncclient:jar:4.1.4
org.apache.httpcomponents:httpclient-cache:jar:4.5.13
org.apache.httpcomponents:httpclient:jar:4.5.13
org.apache.httpcomponents:httpcore-nio:jar:4.4.12
org.apache.httpcomponents:httpcore:jar:4.4.13
org.apache.httpcomponents:httpmime:jar:4.3.6

org.apache.httpcomponents:httpmime:jar:4.5.13
org.apache.logging.log4j:log4j-1.2-api:jar:2.19.0
org.apache.logging.log4j:log4j-api:jar:2.19.0
org.apache.logging.log4j:log4j-core:jar:2.19.0
org.apache.logging.log4j:log4j-jcl:jar:2.19.0
org.apache.logging.log4j:log4j-slf4j2-impl:jar:2.19.0
org.apache.logging.log4j:log4j-web:jar:2.19.0
org.apache.lucene:lucene-analyzers-common:jar:8.11.1
org.apache.lucene:lucene-backward-codecs:jar:8.11.1
org.apache.lucene:lucene-core:jar:8.11.1
org.apache.lucene:lucene-grouping:jar:8.11.1
org.apache.lucene:lucene-highlighter:jar:8.11.1
org.apache.lucene:lucene-join:jar:8.11.1
org.apache.lucene:lucene-memory:jar:8.11.1
org.apache.lucene:lucene-misc:jar:8.11.1
org.apache.lucene:lucene-queries:jar:8.11.1
org.apache.lucene:lucene-queryparser:jar:8.11.1
org.apache.lucene:lucene-sandbox:jar:8.11.1
org.apache.lucene:lucene-spatial3d:jar:8.11.1
org.apache.lucene:lucene-suggest:jar:8.11.1
org.apache.neethi:neethi:jar:3.2.0
org.apache.poi:poi-ooxml-schemas:jar:4.1.2
org.apache.poi:poi-ooxml:jar:4.1.2
org.apache.poi:poi:jar:4.1.2
org.apache.santuario:xmlsec:jar:2.0.5
org.apache.santuario:xmlsec:jar:2.1.5
org.apache.taglibs:taglibs-standard-impl:jar:1.2.5
org.apache.tika:tika-core:jar:1.24.1
org.apache.tomcat:tomcat-api:jar:7.0.109
org.apache.tomcat:tomcat-catalina:jar:7.0.109
org.apache.tomcat:tomcat-el-api:jar:10.1.10
org.apache.tomcat:tomcat-jasper-el:jar:10.1.10
org.apache.tomcat:tomcat-jasper-el:jar:7.0.109
org.apache.tomcat:tomcat-jasper:jar:7.0.109
org.apache.tomcat:tomcat-jsp-api:jar:7.0.109
org.apache.tomcat:tomcat-juli:jar:7.0.109
org.apache.tomcat:tomcat-util:jar:7.0.109
org.apache.velocity:velocity:jar:1.7
org.apache.ws.commons.axiom:axiom-api:jar:1.2.10

org.apache.ws.commons.axiom:axiom-impl:jar:1.2.10
org.apache.ws.commons:XmlSchema:jar:1.1
org.apache.ws.security:wss4j:jar:1.6.19
org.apache.ws.xmlschema:xmlschema-core:jar:2.3.0
org.apache.xbean:xbean-spring:jar:4.12
org.apache.xbean:xbean-spring:jar:4.22
org.apache.xmlbeans:xmlbeans-xpath:jar:2.6.0
org.apache.xmlbeans:xmlbeans:jar:3.1.0
org.aspectj:aspectjrt:jar:1.8.1
org.aspectj:aspectjrt:jar:1.8.8
org.aspectj:aspectjweaver:jar:1.8.8
org.aspectj:aspectjweaver:jar:1.9.7
org.bitbucket.b_c:jose4j:jar:0.9.3
org.bouncycastle:bc-fips:jar:1.0.2.4
org.checkerframework:checker-qual:jar:3.42.0
org.checkerframework:checker-qual:jar:3.8.0
org.codehaus.castor:castor-core:jar:1.4.1
org.codehaus.castor:castor-xml:jar:1.4.1
org.codehaus.groovy.maven.feature:gmaven-feature-api:jar:1.0-rc-3
org.codehaus.groovy.maven.feature:gmaven-feature-support:jar:1.0-rc-3
org.codehaus.groovy.maven.runtime:gmaven-runtime-1.5:jar:1.0-rc-3
org.codehaus.groovy.maven.runtime:gmaven-runtime-api:jar:1.0-rc-3
org.codehaus.groovy.maven.runtime:gmaven-runtime-default:jar:1.0-rc-3
org.codehaus.groovy.maven.runtime:gmaven-runtime-support:jar:1.0-rc-3
org.codehaus.groovy.maven:gmaven-common:jar:1.0-rc-3
org.codehaus.jettison:jettison:jar:1.3.7
org.codehaus.mojo:animal-sniffer-annotations:jar:1.23
org.codehaus.plexus:plexus-utils:jar:3.3.0
org.codehaus.woodstox:stax2-api:jar:3.1.4
org.codehaus.woodstox:woodstox-core-asl:jar:4.4.1
org.codehaus.woodstox:wstx-asl:jar:3.2.0
org.codehaus.woodstox:wstx-asl:jar:3.2.6
org.codehaus.xfire:xfire-aegis:jar:1.2.6
org.codehaus.xfire:xfire-annotations:jar:1.2.5
org.codehaus.xfire:xfire-core:jar:1.2.6
org.codehaus.xfire:xfire-java5:jar:1.2.6
org.codehaus.xfire:xfire-jaxb2:jar:1.2.6
org.codehaus.xfire:xfire-jaxws:jar:1.2.6
org.codehaus.xfire:xfire-spring:jar:1.2.6

org.conscrypt:conscrypt-openjdk-uber:jar:2.5.2
org.cryptacular:cryptacular:jar:1.1.1
org.dbunit:dbunit:jar:2.5.0
org.dom4j:dom4j:jar:2.1.3
org.eclipse.jdt.core.compiler:ecj:jar:4.4.2
org.eclipse.jetty:jetty-continuation:jSar:8.2.0.v20160908
org.eclipse.jetty:jetty-http:jar:8.2.0.v20160908
org.eclipse.jetty:jetty-io:jar:8.2.0.v20160908
org.eclipse.jetty:jetty-security:jar:8.2.0.v20160908
org.eclipse.jetty:jetty-server:jar:8.2.0.v20160908
org.eclipse.jetty:jetty-servlet:jar:8.2.0.v20160908
org.eclipse.jetty:jetty-util:jar:8.2.0.v20160908
org.eclipse.jetty:jetty-webapp:jar:8.2.0.v20160908
org.eclipse.jetty:jetty-xml:jar:8.2.0.v20160908
org.eclipse.jetty:test-jetty-servlet:jar:8.2.0.v20160908
org.eclipse.jgit:org.eclipse.jgit:jar:4.6.1.201703071140-r
org.eclipse.parsson:parsson:jar:1.0.0
org.elasticsearch.client:elasticsearch-rest-client:jar:7.17.3
org.elasticsearch:elasticsearch-cli:jar:7.17.3
org.elasticsearch:elasticsearch-core:jar:7.17.3
org.elasticsearch:elasticsearch-geo:jar:7.17.3
org.elasticsearch:elasticsearch-lz4:jar:7.17.3
org.elasticsearch:elasticsearch-plugin-classloader:jar:7.17.3
org.elasticsearch:elasticsearch-secure-sm:jar:7.17.3
org.elasticsearch:elasticsearch-x-content:jar:7.17.3
org.elasticsearch:elasticsearch:jar:7.17.3
org.fusesource.hawtbuf:hawtbuf:jar:1.11
org.glassfish.jaxb:jaxb-runtime:jar:2.3.1
org.glassfish.jaxb:jaxb-runtime:jar:2.3.2
org.glassfish.jaxb:txw2:jar:2.3.1
org.glassfish.jaxb:txw2:jar:2.3.2
org.glassfish:javax.el:jar:3.0.0
org.glassfish:javax.json:jar:1.0.4
org.hamcrest:hamcrest-core:jar:1.1
org.hamcrest:hamcrest-core:jar:1.3
org.hamcrest:hamcrest-core:jar:1.4-atlassian-1
org.hamcrest:hamcrest-core:jar:2.2
org.hamcrest:hamcrest-library:jar:1.3
org.hamcrest:hamcrest-library:jar:1.4-atlassian-1

org.hamcrest:hamcrest-library:jar:2.2
org.hamcrest:hamcrest:jar:2.2
org.hdrhistogram:HdrHistogram:jar:2.1.9
org.hibernate.common:hibernate-commons-annotations:jar:5.1.2.Final
org.hibernate.validator:hibernate-validator:jar:6.0.20.Final
org.hibernate.validator:hibernate-validator:jar:6.0.22.Final
org.hibernate:hibernate-core:jar:5.4.26.Final
org.hibernate:hibernate-ehcache:jar:5.4.26.Final
org.javassist:javassist:jar:3.19.0-GA
org.javassist:javassist:jar:3.27.0-GA
org.jboss.logging:jboss-logging:jar:3.3.2.Final
org.jboss.logging:jboss-logging:jar:3.4.1.Final
org.jboss.spec.javax.transaction:jboss-transaction-api_1.2_spec:jar:1.1.1.Final
org.jboss:jandex:jar:2.1.3.Final
org.jdom:jdom2:jar:2.0.6
org.jetbrains.kotlin:kotlin-stdlib-common:jar:1.6.20
org.jetbrains.kotlin:kotlin-stdlib-jdk7:jar:1.6.20
org.jetbrains.kotlin:kotlin-stdlib-jdk8:jar:1.6.20
org.jetbrains.kotlin:kotlin-stdlib:jar:1.6.20
org.jetbrains:annotations:jar:13.0
org.json:json:jar:20090211
org.json:json:jar:20231013
org.jsoup:jsoup:jar:1.10.3
org.jvnet.jaxb2_commons:jaxb2-basics-runtime:jar:0.9.5
org.jvnet.staxex:stax-ex:jar:1.8
org.jvnet.staxex:stax-ex:jar:1.8.1
org.ldaptive:ldaptive:jar:1.0.9
org.liquibase:liquibase-core:jar:4.8.0
org.lz4:lz4-java:jar:1.8.0
org.mapstruct:mapstruct:jar:1.2.0.Final
org.mariadb.jdbc:mariadb-java-client:jar:2.7.12
org.mockito:mockito-all:jar:1.9.5
org.mockito:mockito-core:jar:1.9.5
org.mockito:mockito-core:jar:5.4.0
org.objenesis:objenesis:jar:1.0
org.objenesis:objenesis:jar:3.3
org.opensaml:opensaml-core:jar:3.3.0
org.opensaml:opensaml-messaging-api:jar:3.3.0
org.opensaml:opensaml-messaging-impl:jar:3.3.0

org.opensaml:opensaml-profile-api:jar:3.3.0
org.opensaml:opensaml-profile-impl:jar:3.3.0
org.opensaml:opensaml-saml-api:jar:3.3.0
org.opensaml:opensaml-saml-impl:jar:3.3.0
org.opensaml:opensaml-security-api:jar:3.3.0
org.opensaml:opensaml-security-impl:jar:3.3.0
org.opensaml:opensaml-soap-api:jar:3.3.0
org.opensaml:opensaml-soap-impl:jar:3.3.0
org.opensaml:opensaml-storage-api:jar:3.3.0
org.opensaml:opensaml-storage-impl:jar:3.3.0
org.opensaml:opensaml-xacml-api:jar:3.3.0
org.opensaml:opensaml-xacml-impl:jar:3.3.0
org.opensaml:opensaml-xacml-saml-api:jar:3.3.0
org.opensaml:opensaml-xacml-saml-impl:jar:3.3.0
org.opensaml:opensaml-xmlsec-api:jar:3.3.0
org.opensaml:opensaml-xmlsec-impl:jar:3.3.0
org.ow2.asm:asm:jar:5.0.4
org.ow2.asm:asm:jar:9.2
org.owasp.encoder:encoder-jsp:jar:1.2.3
org.owasp.encoder:encoder:jar:1.2.3
org.postgresql:postgresql:jar:42.7.3
org.quartz-scheduler:quartz-jobs:jar:2.3.2
org.quartz-scheduler:quartz:jar:2.3.2
org.reactivestreams:reactive-streams:jar:1.0.4
org.reflections:reflections:jar:0.9.9-RC1
org.samba.jcifs:jcifs:jar:1.3.3
org.skyscreamer:jsonassert:jar:1.2.3
org.slf4j:jcl-over-slf4j:jar:1.7.35
org.slf4j:jul-to-slf4j:jar:1.7.12
org.slf4j:slf4j-api:jar:1.7.35
org.slf4j:slf4j-api:jar:2.0.7
org.smali:baksmali:jar:2.0.3
org.smali:dexlib2:jar:2.0.3
org.smali:smali:jar:2.0.3
org.smali:util:jar:2.0.3
org.springframework.boot:spring-boot-autoconfigure:jar:2.7.18
org.springframework.boot:spring-boot:jar:2.7.18
org.springframework.data:spring-data-commons:jar:2.2.2.RELEASE
org.springframework.data:spring-data-keyvalue:jar:2.2.2.RELEASE

org.springframework.data:spring-data-redis:jar:2.2.2.RELEASE
org.springframework.hateoas:spring-hateoas:jar:0.24.0.RELEASE
org.springframework.plugin:spring-plugin-core:jar:1.2.0.RELEASE
org.springframework.plugin:spring-plugin-metadata:jar:1.2.0.RELEASE
org.springframework.security.extensions:spring-security-kerberos-core:jar:1.0.0.M2
org.springframework.security.oauth:spring-security-oauth2:jar:2.3.6.RELEASE
org.springframework.security:spring-security-config:jar:5.7.5
org.springframework.security:spring-security-core:jar:5.7.5
org.springframework.security:spring-security-crypto:jar:5.3.12.RELEASE
org.springframework.security:spring-security-crypto:jar:5.7.5
org.springframework.security:spring-security-jwt:jar:1.0.9.RELEASE
org.springframework.security:spring-security-test:jar:5.7.5
org.springframework.security:spring-security-web:jar:5.7.5
org.springframework.social:spring-social-config:jar:1.1.6.RELEASE
org.springframework.social:spring-social-core:jar:1.1.6.RELEASE
org.springframework.social:spring-social-test:jar:1.0.3.RELEASE
org.springframework.social:spring-social-web:jar:1.1.6.RELEASE
org.springframework.ws:spring-ws-core:jar:3.0.7.RELEASE
org.springframework.ws:spring-ws-core:jar:3.1.6
org.springframework.ws:spring-xml:jar:3.0.7.RELEASE
org.springframework.ws:spring-xml:jar:3.1.6
org.springframework:spring-aop:jar:5.3.35
org.springframework:spring-aspects:jar:5.3.35
org.springframework:spring-beans:jar:5.3.33
org.springframework:spring-beans:jar:5.3.35
org.springframework:spring-context-support:jar:5.3.35
org.springframework:spring-context:jar:5.3.35
org.springframework:spring-core:jar:5.3.33
org.springframework:spring-core:jar:5.3.35
org.springframework:spring-expression:jar:5.3.35
org.springframework:spring-jcl:jar:5.3.33
org.springframework:spring-jcl:jar:5.3.35
org.springframework:spring-jdbc:jar:5.3.35
org.springframework:spring-jms:jar:5.3.33
org.springframework:spring-orm:jar:5.3.35
org.springframework:spring-oxm:jar:5.3.35
org.springframework:spring-test:jar:5.3.33
org.springframework:spring-test:jar:5.3.35
org.springframework:spring-tx:jar:5.3.35

org.springframework:spring-web:jar:5.3.33
org.springframework:spring-web:jar:5.3.35
org.springframework:spring-webmvc:jar:5.3.35
org.testng:testng:jar:6.10
org.threeten:threetenbp:jar:1.6.8
org.tinyradius:tinyradius:jar:1.0
org.togglz:togglz-core:jar:2.4.1.Final
org.togglz:togglz-junit:jar:2.4.1.Final
org.togglz:togglz-servlet:jar:2.4.1.Final
org.togglz:togglz-slf4j:jar:2.4.1.Final
org.togglz:togglz-spring-core:jar:2.4.1.Final
org.togglz:togglz-spring-web:jar:2.4.1.Final
org.togglz:togglz-testing:jar:2.4.1.Final
org.tuckey:urlrewritefilter:jar:3.1.0
org.xerial:sqlite-jdbc:jar:3.28.0
org.xmlunit:xmlunit-core:jar:2.1.1
org.xmlunit:xmlunit-legacy:jar:2.1.1
org.yaml:snakeyaml:jar:1.26
oro:oro:jar:2.0.8
p6spy:p6spy:jar:1.3
redis.clients:jedis:jar:3.1.0
software.amazon.ion:ion-java:jar:1.0.2
stax:stax-api:jar:1.0.1
wsdl4j:wsdl4j:jar:1.6.1
xalan:xalan:jar:2.7.3
xerces:xercesImpl:jar:2.12.1
xfire:xfire-jsr181-api:jar:1.0-M1
xml-apis:xml-apis:jar:1.4.01
xml-resolver:xml-resolver:jar:1.2
xmlpull:xmlpull:jar:1.1.3.1
xmlunit:xmlunit:jar:1.5
xpp3:xpp3:jar:1.1.4c