Comment: Comment-1-
Comment: Comment-2-
Comment: Comment-3-
Comment: Comment-4-
Comment: Comment-5-
Comment: Comment-6-
Comment: Comment-7-
Comment: Comment-8-

Protection Profile for Mobile Device Management

Version: 5.0
2024-11-15
National Information Assurance Partnership

Revision History

Version	Date	Comment
1.0	2013-10-21	Initial Release
1.1	2014-02-07	Typographical changes and clarifications to front-matter
2.0	2014-12-31	Separation of MDM agent SFRs Updated cryptography, protocol, X.509 requirements. Updated management functions to match MDF PP v2.0. Included SSH as a remote administration protocol. Removed IPsec as protocol to communicate to MDM agent. Added X509 enrollment objective requirement. Added Optional Mobile Application Store requirements.
3.0	2016-11-21	Updates to align with Technical Decisions Added requirements to support BYOD use case Removed IPsec and SSH requirements, which are now contained in EPs
4.0	2018-09-24	Updates to align with Technical Decisions Removed platform dependency Removed TLS SFRs and use the TLS Functional Package Allowed for a distributed TOE
5.0	2024-11-15	Updates to align with Technical Decisions Updates to align with CC:2022

Contents

1 Introduction

1.1 Overview

Mobile Device Management (MDM) products allow enterprises to apply security policies to managed devices (MDs), such as smartphones, tablets, and workstations. The purpose of these policies is to establish a security posture adequate to permit MDs to process enterprise data and connect to enterprise network resources.

This document provides a baseline set of Security Functional Requirements (SFRs) for an MDM system, which is the Target of Evaluation (TOE). The MDM system is only one component of an enterprise deployment of MDs. Other components, such as the MD platforms, which enforce the security policies and network access control servers are out of scope.

1.2 Terms

The following sections list Common Criteria and technology terms used in this document.

1.2.1 Common Criteria Terms

Assurance	Grounds for confidence that a TOE meets the SFRs [CC].
Base Protection Profile (Base-PP)	Protection Profile used as a basis to build a PP-Configuration.
Collaborative Protection Profile (cPP)	A Protection Profile developed by international technical communities and approved by multiple schemes.
Common Criteria (CC)	Common Criteria for Information Technology Security Evaluation (International Standard ISO/IEC 15408).
Common Criteria Testing Laboratory	Within the context of the Common Criteria Evaluation and Validation Scheme (CCEVS), an IT security evaluation facility accredited by the National Voluntary Laboratory Accreditation Program (NVLAP) and approved by the NIAP Validation Body to conduct Common Criteria-based evaluations.
Common Evaluation Methodology (CEM)	Common Evaluation Methodology for Information Technology Security Evaluation.
Distributed TOE	A TOE composed of multiple components operating as a logical whole.
Extended Package (EP)	A deprecated document form for collecting SFRs that implement a particular protocol, technology, or functionality. See Functional Packages.
Functional Package (FP)	A document that collects SFRs for a particular protocol, technology, or functionality.
Operational Environment (OE)	Hardware and software that are outside the TOE boundary that support the TOE functionality and security policy.
Protection Profile (PP)	An implementation-independent set of security requirements for a category of products.
Protection Profile Configuration (PP-Configuration)	A comprehensive set of security requirements for a product type that consists of at least one Base-PP and at least one PP-Module.
Protection Profile Module (PP-Module)	An implementation-independent statement of security needs for a TOE type complementary to one or more Base-PPs.
Security Assurance Requirement (SAR)	A requirement to assure the security of the TOE.

Security Functional Requirement (SFR)	A requirement for security enforcement by the TOE.
Security Target (ST)	A set of implementation-dependent security requirements for a specific product.
Target of Evaluation (TOE)	The product under evaluation.
TOE Security Functionality (TSF)	The security functionality of the product under evaluation.
TOE Summary Specification (TSS)	A description of how a TOE satisfies the SFRs in an ST.

1.2.2 Technical Terms

API Application Programming Interface (API)	A specification of routines, data structures, object classes, and variables that allows an application to make use of services provided by another software component, such as a library. APIs are often provided for a set of libraries included with the platform.
Administrator	The person who is responsible for management activities, including setting the policy that is applied by the enterprise on the MD.
Critical Security Parameter (CSP)	Security-related information whose disclosure or modification can compromise the security of a cryptographic module or authentication system.
Data	Program or application or data files that are stored or transmitted by a server or MD.
Data Encryption Key (DEK)	A key used to encrypt data-at-rest.
Developer Modes	States in which additional services are available to a user in order to provide enhanced system access for debugging of software.
Enrolled State	The state in which a MD is administered and configured by a policy from an MDM.
Enrollment over Secure Transport (EST)	Cryptographic protocol that describes an X.509 certificate management protocol targeting public key infrastructure (PKI) clients that need to acquire client certificates and associated certificate authority (CA) certificates.
Enterprise Applications	Applications that are provided and managed by the enterprise as opposed to a public application store.
Enterprise Data	Any data residing in enterprise servers or temporarily stored on MDs to which the MD user is allowed access according to the security policy defined by the enterprise and implemented by the administrator.
Key Encryption Key (KEK)	A key that is used to encrypt other keys, such as DEKs or storage repositories that contain keys.
Locked State	Device state where the device is powered on but most functionality is unavailable for use without authentication.
Managed Device (MD)	A device enrolled and managed, or to be enrolled and managed, by an MDM system. This device is a mobile device typically evaluated under the MDF PP or a device utilizing a GPOS typically evaluated under the GPOS PP.
Managed Device User	The person who uses and is held responsible for an MD.
Mobile Application	An MDM feature that allows for an organization to substitute a device manufacturer's application store for one that restricts what applications are made available to users.

Store (MAS)	
Mobile Device	A device which is composed of a hardware platform and its system software. The device typically provides wireless connectivity and may include software for functions like secure messaging, email, web, VPN connection, and VoIP (Voice over IP), for access to the protected enterprise network, enterprise data and applications, and for communicating to other MDs.
Mobile Device Management (MDM)	Products that allow enterprises to apply security policies to MDs. This system consists of two primary components: the MDM server and the MDM agent.
Operating System (OS)	Software which runs at the highest privilege level and can directly control hardware resources. Modern mobile devices typically have at least two primary operating systems: one which runs on the cellular baseband processor and one which runs on the application processor. The platform of the application processor handles most user interaction and provides the execution environment for apps. The platform of the cellular baseband processor handles communications with the cellular network and may control other peripherals. The term OS, without context, may be assumed to refer to the platform of the application processor.
Powered-Off State	MD shutdown state.
Protected Data	All non-TSF data on the MD, including user or enterprise data. Protected data is encrypted while the MD is in the powered-off state. This includes keys in software-based storage. May overlap with sensitive data.
Root Encryption Key (REK)	A key tied to a particular device that is used to encrypt all other keys for that device.
Sensitive Data	Data that is encrypted by the MD. May include all user or enterprise data or may be data for specific applications such as emails, messaging, documents, calendar items, or contacts. May be protected while the MD is in the locked state. Must include at minimum some keys in software-based key storage.
Trust Anchor Database	A list of trusted root Certificate Authority certificates.
Unenrolled State	Device state when it is not managed by an MDM.
Unlocked State	Device state where it is powered on and the user has been authenticated. The device functionality is available for use by the user.

1.3 Compliant Targets of Evaluation

The Mobile Device Management (MDM) system consists of two primary components: the MDM server software and the MDM agent. Optionally, the MDM system may consist of a separate Mobile Application Store (MAS) server.

1.3.1 TOE Boundary

The MDM system operational environment consists of the MD on which the MDM agent resides, the platform on which the MDM server runs, and an untrusted wireless network over which they communicate, as pictured below.

Figure 1: MDM System Operating Environment

The **MDM server** is software (an application, service, etc.) on a general-purpose platform, a network device, or cloud architecture executing in a trusted network environment. The MDM server provides administration of MD policies and reporting on the device behavior. The MDM server is responsible for managing device

enrollment, configuring and sending policies to the MDM agents, collecting reports on device status, and sending commands to the agents. The MDM server may be standalone or distributed, where a distributed TOE is one that requires multiple distinct components to operate as a logical whole in order to fulfill the requirements of this PP (a more extensive description of distributed MDMs is given in section 3).

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

The **Mobile Application Store (MAS)** hosts applications for the enterprise, authenticates agents, and securely transmits applications to enrolled MDs. The MAS functionality can be included as part of the MDM server Software or can be logically distinct. If the MAS functionality is on a physically separate server, then the TOE is distributed with the MDM server and MAS server being separate components.

1.4 Use Cases

This PP defines four use cases:

[USE CASE 1] Enterprise-owned device for general-purpose enterprise use
An enterprise-owned device for general-purpose business use is commonly called Corporate-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 MDs prior to user issuance. Users may use internet connectivity to browse the web, 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 MD status. The MDM may issue commands for remediation actions.

[USE CASE 2] Enterprise-owned device for specialized, high-security use
An enterprise-owned device with intentionally limited network connectivity, tightly controlled configuration, and limited software inventory is appropriate for specialized, high-security use cases. As in the previous use case, the MDM product is used to establish such policies on MDs prior to issuance to users. The device may not be permitted connectivity to any external peripherals. It may only be able to communicate via its Wi-Fi or cellular radios with the enterprise-run network, which may not even permit connectivity to the internet. Use of the device may require compliance with usage policies that are more restrictive than those in any general-purpose use case, yet may mitigate risks to highly sensitive information. Based on the operational environment and the acceptable risk level of the enterprise, those security functional requirements outlined in Section 5 of this PP along with the selections in the Use Case 2 template defined in Appendix G are sufficient for the high-security use case.

[USE CASE 3] Personally-owned device for personal and enterprise use
A personally-owned device which is used for both personal activities and enterprise data is commonly called Bring Your Own Device (BYOD). The device may be provisioned for access to enterprise resources after significant personal usage has occurred. Unlike in the enterprise-owned cases, the enterprise is limited in what security policies it can enforce because the user purchased the device primarily for personal use and is unlikely to accept policies that limit the functionality of the device.

However, because the enterprise allows the user full (or nearly full) access to the enterprise network, the enterprise will require certain security policies, for example a password or screen lock policy and health reporting, such as the integrity of the MD system software, before allowing access. The administrator of the MDM can establish remediation actions, such as wipe of the enterprise data, for non-compliant devices. These controls could potentially be enforced by a separation mechanism built-in to the device itself to distinguish between enterprise and personal activities, or by a third-party application that provides access to enterprise resources and leverages security capabilities provided by the MD. Based on the operational environment and the acceptable risk level of the enterprise, those security functional requirements outlined in Section 5 of this PP along with the selections in the Use Case 3 template defined in Appendix G are sufficient for the secure implementation of this BYOD use case.

[USE CASE 4] Personally-owned device for personal and limited enterprise use
A personally-owned device may also be given access to limited enterprise services such as enterprise email. The enterprise may not need to enforce any security policies on this device because the user does not have full access to the enterprise or enterprise data. However, the enterprise may want secure email and web browsing with assurance that the services being provided to those clients by the device are not compromised. Based on the operational environment and the acceptable risk level of the enterprise, those security functional requirements outlined in Section 5 of this PP are sufficient for the secure implementation of this BYOD use case.

2 Security Problem Definition

2.1 Threats

T.MALICIOUS_APPS
Malicious or flawed application threats exist because apps loaded onto aMD may include malicious or exploitable code. An administrator of the MDM or MD usermay inadvertently import malicious code, or an attacker may insert malicious code into the TOE, resulting in thecompromise of TOE or TOE data.

T.NETWORK_ATTACK
An attacker may masquerade as an MDM server and attempt tocompromise the integrity of the MD by sending malicious management commands.

T.NETWORK_EAVESDROP
An attacker is positioned on a communications channel or elsewhere on thenetwork infrastructure. Attackers may monitor and gain access to data exchangedbetween the MDM server and other endpoints.

T.PHYSICAL_ACCESS
The MD may be lost or stolen, and an unauthorized individual mayattempt to access user data. Although these attacks are primarily directed against theMD platform, the TOE configures features on MDs that mitigate the this threat against them. Therefore, misuse or failure of the TOE may also compromise the MD, exposing it to this threat.

2.2 Assumptions

A.COMPONENTS_RUNNING
This assumption applies to distributed TOEs only. For distributed TOEs, it is assumed that the availability of all TOE components is checked as appropriate to reduce the risk of an undetected attack on (or failure of) one or more TOE components. It is also assumed that in addition to the availability of all components it is also checked as appropriate that the audit functionality is running properly on all TOE components.

A.CONNECTIVITY
The TOE relies on network connectivity to carry out its management activities. The TOE will robustly handle instances when connectivity is unavailable or unreliable.

A.MDM_SERVER_PLATFORM
The MDM server relies on 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
One or more competent, trusted personnel who are not careless, willfully negligent, or hostile, are assigned and authorized as the TOE Administrators, and do so using and abiding by guidance documentation.

A.PROPER_USER
MD users are not willfully negligent or hostile, and use the device within compliance of a reasonable enterprise security policy.

2.3 Organizational Security Policies

P.ACCOUNTABILITY
Personnel operating the TOE shall be accountable for their actions within the TOE.

P.ADMIN
The configuration of the MD security functions must adhere to the enterprise security policy.

P.DEVICE_ENROLL
A MD must be enrolled for a specific user by the administrator of the MDM prior to being used in the enterprise network by the user.

P.NOTIFY
The MD user must immediately notify the administrator if a MD is lost or stolen so that the administrator may apply remediation actions via the MDM system.

3 Security Objectives

3.1 Security Objectives for the Operational Environment

OE.COMPONENTS_RUNNING
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.IT_ENTERPRISE
The enterprise IT infrastructure provides security for a network that is available to the TOE and MDs that prevents unauthorized access.

OE.PROPER_ADMIN
TOE Administrators are trusted to follow and apply all administrator guidance in a trusted manner.

OE.PROPER_USER
Users of the MD are trained to securely use the MD and apply all guidance in a trusted manner.

OE.TIMESTAMP
Reliable timestamps is provided by the operational environment for the TOE.

OE.WIRELESS_NETWORK
A wireless network will be available to the MDs.

3.2 Security Objectives Rationale

This section describes how the assumptions, threats, and organizational security policies map to the security objectives.

Table 1: Security Objectives Rationale

Threat, Assumption, or OSP	Security Objectives	Rationale
A.COMPONENTS_RUNNING	OE.COMPONENTS_RUNNING	The operational environment objective OE.COMPONENTS_RUNNING is realized through A.COMPONENTS_RUNNING.
A.CONNECTIVITY	OE.WIRELESS_NETWORK	The operational environment objective OE.WIRELESS_NETWORK is realized through A.CONNECTIVITY.
A.MDM_SERVER_PLATFORM	OE.TIMESTAMP	The operational environment objective OE.TIMESTAMP is realized through A.MDM_SERVER_PLATFORM.
A.PROPER_ADMIN	OE.PROPER_ADMIN	The operational environment objective OE.PROPER_ADMIN is realized through A.PROPER_ADMIN.
A.PROPER_USER	OE.PROPER_USER	The operational environment objective OE.PROPER_USER is realized through A.PROPER_USER.
P.ADMIN	OE.PROPER_ADMIN	The organizational security policy P.ADMIN is realized through OE.PROPER_ADMIN.
P.DEVICE_ENROLL	OE.IT_ENTERPRISE	The organizational security policy P.DEVICE_ENROLL is realized through OE.IT_ENTERPRISE.
P.NOTIFY	OE.PROPER_USER	The organizational security policy P.NOTIFY is realized through OE.PROPER_USER.

4 Security Requirements

This chapter describes the security requirements which have to be fulfilled by the product under evaluation. Those requirements comprise functional components from Part 2 and assurance components from Part 3 of [CC]. The following conventions are used for the completion of operations:

- **Refinement** operation (denoted by **bold text** or ~~strikethrough text~~): Is used to add details to a requirement or to remove part of the requirement that is made irrelevant through the completion of another operation, and thus further restricts a requirement.
- **Selection** (denoted by *italicized text*): Is used to select one or more options provided by the [CC] in stating a requirement.
- **Assignment** operation (denoted by *italicized text*): Is used to assign a specific value to an unspecified parameter, such as the length of a password. Showing the value in square brackets indicates assignment.
- **Iteration** operation: Is indicated by appending the SFR name with a slash and unique identifier suggesting the purpose of the operation, e.g. "/EXAMPLE1."

For a distributed TOE, the ST author should reference Table 6 for guidance on how each SFR should be met. The table details whether SFRs should be met by all TOE components, by at least one TOE component or whether they are dependent on the feature being implemented by the TOE component. The ST for a distributed TOE must include a mapping of SFRs to each of the components of the TOE. (Note that this deliverable is examined as part of the ASE_TSS.1 and AVA_VAN.1 Evaluation Activities.) **Test Environment for Evaluation Activities**

As described in the evaluation activities below, the ST for an MDM system is required to list all the supported MDM agents and MD platforms with which an MDM server operates. The identified evaluation activities for testing that includes the use of an agent should be completed for each MDM agent and platform listed in the ST.

The evaluator's activities for testing that include use of an agent will ensure that the Server interacts appropriately with the agent (i.e., sends a policy update to the agent), but will not ensure that the agent handles the received data correctly (i.e., appropriately applies the policy to the device), as that is accounted for in the Evaluation Activities in the Mobile Device Fundamentals PP or the PP-Module for Mobile Device Management Agents.

4.1 Security Functional Requirements

4.1.1 Auditable Events for Mandatory SFRs

Table 2: Auditable Events for Mandatory Requirements

Requirement	Auditable Events	Additional Audit Record Contents
FAU_ALT_EXT.1	Type of alert	Identity of MD that sent alert.
FAU_GEN.1/AUDITGEN	No events specified	N/A
FAU_NET_EXT.1	No events specified	N/A
FAU_STG.1	No events specified	N/A
FCS_CKM.1	[**selection, choose one of**: *Failure of key generation activity for authentication keys*, *none*]	No additional information
FCS_CKM.2	No events specified	N/A
FCS_CKM.6	No events specified	N/A
FCS_COP.1/CONF_ALG	No events specified	N/A
FCS_COP.1/HASH_ALG	No events specified	N/A
FCS_COP.1/KEY_HASH	No events specified	N/A
FCS_COP.1/SIGN_ALG	No events specified	N/A
FCS_RBG.1	Failure of the randomization process.	None.
FCS_STG_EXT.1	No events specified	N/A

FIA_CLI_EXT.1	No events specified	N/A
FIA_ENR_EXT.1	Failure of MD user authentication	Presented username
FIA_UAU.1	No events specified	N/A
FIA_X509_EXT.1/CERTVAL_MAN	Failure to validate X.509 certificate	Reason for failure
FIA_X509_EXT.2	Failure to establish connection to determine revocation status.	No additional information
FMT_MOF.1/FUNCBE	Issuance of command to perform function	Command sent and identity of MDM agent recipients
	Change of policy settings	Policy changed and value or full policy
FMT_MOF.1/MANAGEMENT_ENROLL	Enrollment by a user	Identity of user
FMT_POL_EXT.1	No events specified	N/A
FMT_SMF.1/SERVER_CONF_AGENT	No events specified	N/A
FMT_SMF.1/SERVER_CONF_SERVER	Success or failure of function	No additional information
FMT_SMR.1/SECMAN_ROLES	No events specified	N/A
FPT_API_EXT.1	No events specified	N/A
FPT_FLS.1	Failure of the TSF.	None.
FPT_LIB_EXT.1	No events specified	N/A
FPT_TST.1	Execution of self-tests.	None.
FPT_TST_EXT.1	Initiation of self-test	No additional information
	Failure of self-test	Algorithm that caused failure
	Detected integrity violation	The TSF code file that caused the integrity violation
FPT_TUD_EXT.1	Success or failure of signature verification	No additional information
FTP_ITC.1/INTER_XFER_IT	Initiation and termination of the trusted channel	• Trusted channel protocol • Non-TOE endpoint of connection
FTP_ITC_EXT.1	No events specified	N/A
FTP_TRP.1/TRUSTPATH_ENROLL	Initiation and termination of the trusted channel	Trusted channel protocol
FTP_TRP.1/TRUSTPATH_REM_ADMIN	Initiation and termination of the trusted channel	• Trusted channel protocol • Identity of administrator

4.1.2 Security Audit (FAU)

FAU_ALT_EXT.1 Server Alerts

FAU_ALT_EXT.1.1

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

- Change in enrollment status
- Failure to apply policies to a MD

- [**selection**: *[**assignment**: other events] , no other events*]

Application Note: An alert can be defined as anything providing straightaway notice to the administrator. An alert is different from an audit record, however the fact that an alert was sent should be audited per FAU_GEN.1. Email, pop-up notifications, or other methods are acceptable forms of alerts.

The MDM agent is required to report to the MDM server on successful application of policies on a MD, and failures can be inferred from the absence of such alerts. This requirement is intended to ensure that the MDM server notifies administrators when policies are not properly installed. Failure to properly install policy updates does not affect the enrollment status of the MD.

Evaluation Activities

FAU_ALT_EXT.1
TSS
The evaluator shall examine the TSS and verify that it describes how the alert system is implemented. The evaluator shall also verify that a description of each assigned event is provided in the TSS.

Guidance
The evaluator shall examine the guidance document and verify that it describes how the alerts can be configured, if configurable.

Tests
For each MDM agent or platform listed as supported in the ST:

- *Test FAU_ALT_EXT.1:1: The evaluator shall enroll a device and ensure that the MDM server alerts the administrator of the change in enrollment status. The evaluator shall unenroll (retire) a device and ensure that the MDM server alerts the administrator of the change in enrollment status.*
- *Test FAU_ALT_EXT.1:2: The evaluator shall configure policies, which the MDM agent should not be able to apply. These policies shall include:*
  - *a setting which is configurable on the MDM server interface but not supported by the platform on which the MDM agent runs, if any such settings exist*
  - *a valid configuration setting with an invalid parameter, which may require manual modification of the policy prior to transmission to the device*

  The evaluator shall deploy such policies and verify that the MDM server alerts the administrator about the failed application of the policy.

  Alternate test: The evaluator shall configure a policy that the MDM agent can successfully apply. The evaluator blocks the response from the MDM agent of the successful application of the policy, resulting in the MDM server alerting of a failure of application. The evaluator shall verify that the MDM server alerts the administrator about the failed application of the policy.
- *Test FAU_ALT_EXT.1:3: (Conditional) The evaluator shall trigger each of the events listed and ensure that the MDM server alerts the administrator.*

FAU_CRP_EXT.1 Support for Compliance Reporting of Mobile Device Configuration

This is an objective component.

FAU_CRP_EXT.1.1

The TSF shall provide [**selection**: *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/INTER_XFER_IT. The provided information for each enrolled MD includes:

- The current version of the MD firmware or software
- The current version of the hardware model of the device
- The current version of installed applications
- List of MD configuration policies that are in place on the device (as defined in FMT_SMF.1.1/SERVER_CONF_AGENT)
- [**selection**: *[**assignment**: list of other available information about enrolled devices] , no other information*].

Application Note: The intent of this requirement is that the MDM server be able to provide compliance information about enrolled MDs for use by other

enterprise security infrastructure systems. There are active standards efforts underway by the Internet Engineering Task Force (IETF) Security Automation and Continuous Monitoring (SACM) Working Group and others to define protocols and standards to assess and report on endpoint device posture. We expect that this requirement will evolve in future versions of this Protection Profile as standards efforts mature.

Evaluation Activities

FAU_CRP_EXT.1
TSS
TBD

Guidance
The evaluator shall check to ensure that the operational guidance contains instructions on how to access the MDM server's compliance reporting interface.

Tests
- *Test FAU_CRP_EXT.1:1: Using the operational guidance, the evaluator shall demonstrate the ability to access the compliance reporting interface from an authorized entity and successfully obtain information about enrolled devices.*
- *Test FAU_CRP_EXT.1:2: The evaluator shall attempt to access the compliance reporting interface from an unauthorized entity and demonstrate that the attempt is denied.*

FAU_GEN.1/AUDITGEN Audit Data Generation

FAU_GEN.1.1/AUDITGEN

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

1. All administrative actions
2. [**selection**: *Commands issued to the MDM agent, none*]
3. Specifically defined auditable events listed in Table 2
4. [**selection**:
   - *Start up and shut down of the audit functions*
   - ***Auditable events defined in Table t-audit-optional for Strictly Optional SFRs***
   - ***Auditable events defined in Table t-audit-objective for Objective SFRs***
   - ***Auditable events defined in Table t-audit-sel-based for Selection-Based SFRs***
   - ***Auditable events defined in the audit table for the TLS Functional Package (see Table 3)***
   - *no other events*

   ].

Application Note: This requirement outlines the events for which audit data must be generated by either the MDM system or the MDM server platform. Each audit data event may be written by the MDM system or may be dispatched to the operating system on which it runs. It is acceptable to select both "invoke platform-provided functionality" and "implement functionality." It should be specified which auditable events are completed by the MDM system and which are completed by the MDM platform.

The ST author can include other auditable events in the assignment; they are not limited to the list presented. All audits must contain at least the information mentioned in FAU_GEN.1.2/AUDITGEN, but may contain more information which can be assigned.

For distributed TOEs, each component must generate audit data for each of the SFRs that it implements. If more than one TOE component is involved when an audit event is triggered, the event has to be audited on each component (e.g., rejection of a connection by one component while attempting to establish a secure communication channel between two components should result in an audit event being generated by both components). This is not limited to error cases but also includes events about successful actions like successful build up or tear down of a secure communication channel between TOE components.

Item **a** above requires all administrative actions to be auditable. Administrative actions refer to any management functions specified by FMT_MOF.1/FUNCBE.

Thus no additional specification for the auditability of these actions is specified in Table 2 aside from those that require additional record content. If the TOE is distributed and the given component does not deal with setting the policy applied to the MDM agent, it is acceptable to not have any administrative actions to audit.

Item **b** includes those commands, which may be performed automatically based on triggers or on a schedule. If the TOE component, if distributed, interacts directly with the MDM agent, then "Commands issued to the MDM agent" must be selected. If the TOE component, if distributed, does not interact directly with the MDM agent, then it is acceptable to select "none."

Depending on the specific requirements selected by the ST author from SFR, optional requirements, selection-based requirements, and objective requirements, the ST author should include the appropriate auditable events from Table t-audit-optional, Table t-audit-sel-based, and Table t-audit-objective in the ST for the requirements selected.

In item d above, "start up and shut down of the audit functions" must be selected if the TSF has a start up and shut down phase. Additionally, if the TOE is distributed, this applies to all components. If the TOE is not distributed then MDM system is equivalent to MDM server. If the TSF does not have a distinct start up or shut down phase (e.g., a cloud service), this selection is not required.

The following table contains the events enumerated in the auditable events tables for the TLS Functional Package. Inclusion of these events in the ST is subject to selection above, inclusion of the corresponding SFRs in the ST, and support in the FP as represented by a selection in the FP audit table. This list is included here for reference.

Table 3: Auditable Events for Functional Packages

Requirement	Auditable Events	Additional Audit Data Contents
FCS_TLSC_EXT.1	Failure to establish a TLS session.	Reason for failure.
FCS_TLSC_EXT.1	Failure to verify presented identifier.	Presented identifier and reference identifier.
FCS_TLSS_EXT.1	Failure to establish a TLS session.	Reason for failure.
FCS_DTLSC_EXT.1	Failure of the certificate validity check.	Issuer Name and Subject Name of certificate.
FCS_DTLSS_EXT.1	Failure of the certificate validity check.	Issuer Name and Subject Name of certificate.

FAU_GEN.1.2/AUDITGEN

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

- date and time of the event
- type of event
- subject identity (if applicable)
- and the outcome (success or failure) of the event
- additional information in Table 2
- [**selection**:
  - *additional information defined in Table t-audit-optional for Strictly Optional SFRs*
  - *additional information defined in Table t-audit-objective for Objective SFRs*
  - *additional information defined in Table t-audit-sel-based for Selection-Based SFRs*
  - *additional information defined in the audit table for the TLS Functional Package (see Table 3)*
  - *no other additional information*

  ]
- [**assignment**: *other audit relevant information*].

Application Note: This requirement outlines the information to be included in audit data. All audits must contain at least the information mentioned in

FAU_GEN.1.2/AUDITGEN, but may contain more information which can be assigned. The ST author must identify in the TSS which information of the audit data is performed by the TSF and that which is performed by the TOE platform.

Evaluation Activities

FAU_GEN.1.1/AUDITGEN

TSS

The evaluator shall check the TSS and ensure that it lists all of the auditable events. The evaluator shall check to make sure that every audit event type mandated by the PP is described in the TSS. The evaluator shall verify that for every audit event described in the TSS, the description indicates where the audit event is generated (TSF, TOE platform).

If "invoke platform-provided functionality" is selected, the evaluator shall examine the TSS to verify that it describes (for each supported platform) how this functionality is invoked (it should be noted that this may be through a mechanism that is not implemented by the MDM server; nonetheless, that mechanism will be identified in the TSS as part of this evaluation activity).

Guidance

The evaluator shall check the administrative guide and ensure that it lists all of the auditable events. The evaluator shall check to make sure that every audit event type mandated by the PP is described.

The evaluator shall also make a determination of the administrative actions that are relevant in the context of this PP including those listed in the management section. The evaluator shall examine the administrative guide and make a determination of which administrative commands are related to the configuration (including enabling or disabling) of the mechanisms implemented in the TOE that are necessary to enforce the requirements specified in the PP.

The evaluator shall document the methodology or approach taken while determining which actions in the administrative guide are security relevant with respect to this PP. The evaluator may perform this activity as part of the activities associated with ensuring the AGD_OPE guidance satisfies the requirements.

Tests

The evaluator shall test the TOE's ability to correctly generate audit data by having the TOE generate audit data for the events listed in the provided table and administrative actions. This should include all instances of an event. The evaluator shall test that audit data are generated for the establishment and termination of a channel for each of the cryptographic protocols contained in the ST. For administrative actions, the evaluator shall test that each action determined by the evaluator above to be security relevant in the context of this PP is auditable.

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

FAU_GEN.1.2/AUDITGEN

TSS

The evaluator shall check the TSS and ensure that it provides a format for audit data. Each audit data format type must be covered, along with a brief description of each field.

Guidance

The evaluator shall check the administrative guide and ensure that it provides a format for audit data. Each audit data format type must be covered, along with a brief description of each field. The evaluator shall check to make sure that the description of the fields contains the information required in FAU_GEN.1.2/AUDITGEN.

Tests

When verifying the test results from FAU_GEN.1.1/AUDITGEN, the evaluator shall ensure the audit data generated during testing matches the format specified in the administrative guide, and that the fields for each audit data have the proper entries.

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

FAU_GEN.1/MAS_SERVER Audit Data Generation (MAS Server)

This is a selection-based component. Its inclusion depends upon selection from FMT_MOF.1.1/FUNCBE.

FAU_GEN.1.1/MAS_SERVER

The **MAS Server** shall be able to generate audit data of the following auditable events: [

a. *Failure to push a new application on a MD*
b. *Failure to update an existing application on a MD*

].

Application Note: The MDM agent is required to report to the MAS Server on successful receipt of an application or update on a MD, and failures can be inferred from the absence of such alerts.

FAU_GEN.1.2/MAS_SERVER

The [**selection**: ***MAS Server*** , ***MAS Server platform***] shall record within the audit data at least the following information:

- date and time of the auditable event
- type of event
- MD identity
- [**assignment**: *other audit relevant information*.]

Application Note: All audits must contain at least the information mentioned in FAU_GEN.1.2/MAS_SERVER, but may contain more information which can be assigned. The ST author must identify in the TSS which information of the audit data is performed by the TSF and that which is performed by the TOE platform.

This requirement is claimed if "enable, disable, and modify policies listed in FMT_SMF.1/MAS" is selected in FMT_MOF.1.1/FUNCBE.

Evaluation Activities

FAU_GEN.1/MAS_SERVER
TSS
The evaluator shall check the TSS and ensure that it provides a format for audit data. Each audit data format type must be covered, along with a brief description of each field.

Guidance
The evaluator shall check the administrative guide and ensure that it provides a format for audit data. Each audit data format type must be covered, along with a brief description of each field.

The evaluator shall check to make sure that the description of the fields contains the information required in FAU_STG.2.1.

Tests
The evaluator shall verify that when an application or update push fails, that the audit data generated match the format specified in the guidance and that the fields in each audit data have proper entries.

When verifying the test results from FMT_MOF.1.1/FUNCBE, the evaluator shall ensure the audit data generated during testing match the format specified in the administrative guide, and that the fields in each audit data have the proper entries.

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

FAU_NET_EXT.1 Network Reachability Review

FAU_NET_EXT.1.1

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

Application Note: The MDM server establishes the network connectivity status of enrolled agents using periodic reachability event alerts from the agents according to FAU_ALT_EXT.2.1 in the PP-Module for MDM Agent. This status may be determined by sending an update request from the MDM server which the agent is required to respond to or by using scheduled periodic notifications of connectivity initiated by the MDM agent.

Evaluation Activities

FAU_NET_EXT.1
TSS
The evaluator ensures that the TSS describes how reachability events are implemented, for each supported MD platform. The evaluator verifies that this description clearly indicates who (MDM agent or MDM server) initiates reachability events.

Guidance
The evaluator shall verify that the guidance instructs administrators on the method of determining the network connectivity status of an enrolled agent.

Tests
For each MDM agent or platform listed as supported in the ST:

The evaluator shall configure the MDM agent or platform to perform a network reachability test, both with and without such connectivity and shall ensure that by following the guidance, the evaluator can determine results that reflect both.

FAU_SAR.1 Audit Review

This is an optional component. However, applied modules or packages might redefine it as mandatory.

FAU_SAR.1.1

The TSF shall [**selection**: ***invoke platform-provided functionality*** , ***implement functionality***] **to** provide [*authorized administrators*] with the capability to read [*all audit information*] from the audit data.

FAU_SAR.1.2

The TSF shall [**selection**: ***invoke platform-provided functionality*** , ***implement functionality***] **to** provide the audit data in a manner suitable for the **authorized administrators** to interpret the information.

Application Note: The intent of this requirement is to ensure that the administrator can view and interpret the audit data and to prevent unauthorized users from accessing the logs.

Evaluation Activities

FAU_SAR.1
TSS
If "invoke platform-provided functionality" is selected, the evaluator shall examine the TSS to verify that it describes (for each supported platform) how this functionality is invoked (it should be noted that this may be through a mechanism that is not implemented by the MDM server; nonetheless, that mechanism will be identified in the TSS as part of this evaluation activity).

Guidance
The evaluator shall check the operational guidance and ensure that it describes how the administrator accesses the audit data and describes the format of audit data.

Tests
The evaluator shall attempt to view the audit data as the authorized administrator and verify that the action succeeds. The evaluator shall ensure the audit data generated during testing match the format specified in the administrative guide.

FAU_SEL.1 Security Audit Event Selection

This is an optional component. However, applied modules or packages might redefine it as mandatory.

FAU_SEL.1.1

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

1. [*event type*]
2. [*success of auditable security events*

3. *failure of auditable security events*
4. *[**assignment**: other attributes]*]

Application Note: The intent of this requirement is to identify all criteria that can be selected to trigger an audit event. The ST author must select whether the TSF or the platform maintains the audit data. For the ST author, the assignment is used to list any additional criteria or "none."

Evaluation Activities

FAU_SEL.1
TSS
If "invoke platform-provided functionality" is selected, the evaluator shall examine the TSS to verify that it describes (for each supported platform) how this functionality is invoked (it should be noted that this may be through a mechanism that is not implemented by the MDM server; nonetheless, that mechanism will be identified in the TSS as part of this evaluation activity).

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

Tests
The evaluator shall also perform the following tests:

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

FAU_STG.1 Audit Data Storage Location

FAU_STG.1.1

The TSF shall be able to **transmit the generated audit data to an external IT entity using a trusted channel according to FTP_ITC.1/INTER_XFER_IT** and store generated audit data on the [**selection**: *TOE itself, no other storage*].

Application Note: Per FAU_GEN.1.1/AUDITGEN, audit data may be generated by the TOE, the platform, or both. If audit data is generated by the TOE, the TSF must have the ability to securely transmit this data to a remote entity. It may additionally store this data within the TOE boundary, in which case "the TOE itself" is selected and FAU_STG.2 must be included in the ST. Regardless of whether the TSF stores the audit data it generates within the TOE boundary, the TOE must always be able to securely transmit the audit data it generates to a remote entity. If audit data is generated by the platform, any secure local storage or secure remote transmission of this data is the responsibility of the platform.
Although the audit server is outside of the TOE, the TSF should still be able to support mutual authentication. There are no requirements levied on the audit server, but the TOE should be able to support TLS client certificate authentication. This way if the non-TOE audit server does support verifying client certificates, the TSF is able to make use of that.

For distributed TOEs, each component must be able to export any audit data it generates across a protected channel. This may involve individual components independently transmitting audit data to the same environmental audit server via FTP_ITC.1/INTER_XFER_IT, or it may involve one component of a distributed TOE aggregating audit data generated by other components prior to external transmission. In this case, the internal communication of audit data is protected via the mechanisms specifed in FPT_ITT.1/INTER_XFER or FPT_ITT.1/INTER_XFER_AGENT as appropriate. The intent of this requirement in the context of a distributed TOE is that all audit data generated by all components of the TSF be transmitted to an environmental entity entirely through trusted channels, regardless of whether it is transmitted directly from the TSF to the operational environment or whether it is first transmitted to

another part of the TOE.

Evaluation Activities

FAU_STG.1
TSS
The evaluator shall examine the TSS to ensure it describes the means by which the audit data are transferred to the external audit server, and how the trusted channel is provided.

Guidance
The evaluator shall also examine the operational guidance to determine that it describes the relationship between the local audit data and the audit data that are sent to the audit log server. For example, when an audit event is generated, is it simultaneously sent to the external server and the local store, or is the local store used as a buffer and "cleared" periodically by sending the data to the audit server.

The evaluator shall also examine the operational guidance to ensure it describes how to establish the trusted channel to the audit server, as well as describe any requirements on the audit server (particular audit server protocol, version of the protocol required, etc.), as well as configuration of the TOE needed to communicate with the audit server.

Tests
Testing of the trusted channel mechanism will be performed as specified in the associated evaluation activities for the particular trusted channel mechanism.

The evaluator shall perform the following test for this requirement:

- *Test FAU_STG.1:1: The evaluator shall establish a session between the TOE and the audit server according to the configuration guidance provided. The evaluator shall then examine the traffic that passes between the audit server and the TOE during several activities of the evaluator's choice designed to generate audit data to be transferred to the audit server. The evaluator shall observe that these data are not able to be viewed in the clear during this transfer, and that they are successfully received by the audit server. The evaluator shall record the particular software (name, version) used on the audit server during testing.*

FAU_STG.2 Audit Event Storage

This is a selection-based component. Its inclusion depends upon selection from FAU_STG.1.1.

FAU_STG.2.1

The TSF shall [**selection**: ***invoke platform-provided functionality*** , ***implement functionality***] to protect the stored audit data in the audit trail from unauthorized **modification**.

Application Note: If "store audit data locally" is selected in FAU_STG.1.1, this SFR must be included in the ST.

The purpose of this requirement is to ensure that audit data are stored securely. The ST author is responsible for selecting whether audit data are maintained when audit storage or failure occurs. The ST author must choose a means by which audit data are saved and select the events during which the data will be saved. The TSF may rely on the underlying operating system for this functionality, and the first selection should be made appropriately.

FAU_STG.2.2

The TSF shall be able to [**selection, choose one of**: *prevent*, *detect*] unauthorized modifications to the stored audit data in the audit trail.

Evaluation Activities

FAU_STG.2
TSS
If "invoke platform-provided functionality" is selected, the evaluator shall examine the TSS to verify that it describes (for each supported platform) how the audit data protection functionality is invoked (it should be noted that this may be through a mechanism that is not implemented by the MDM server; nonetheless, that mechanism will be identified in the TSS as part of this evaluation activity).

If "implement functionality" is selected, the evaluator shall ensure that the TSS describes how the audit data are protected from unauthorized modification or deletion. The evaluator shall ensure that the TOE uses audit trail specific protection mechanisms.

Guidance
TBD

Tests
The evaluator shall perform the following tests:

- *Test FAU_STG.2:1: The evaluator shall access the audit trail as an unauthorized user and attempt to modify and delete the audit data. The evaluator shall verify that these attempts fail.*
- *Test FAU_STG.2:2: The evaluator shall access the audit trail as an authorized user and attempt to modify and delete the audit data. The evaluator shall verify that these attempts succeed. The evaluator shall verify that only the records intended for modification and deletion are modified and deleted.*

4.1.3 Communication (FCO)

FCO_CPC_EXT.1 Component Registration Channel Definition

This is an objective component.

FCO_CPC_EXT.1.1

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

- *A channel that meets the secure channel requirements in [**selection**: FTP_ITC.1, FPT_ITT.1/INTER_XFER, FPT_ITT.1/INTER_XFER_AGENT]*
- *A channel that meets the secure registration channel requirements in [**selection**: FTP_TRP.1/TRUSTPATH_ENROLL, FTP_TRP.1/TRUSTPATH_JOIN]*
- *No channel*

] for at least TSF data.

FCO_CPC_EXT.1.3

The TSF shall [**selection**: *invoke platform-provided functionality*, *implement functionality*] to enable an administrator to disable communications between any pair of TOE components.

Application Note: This SFR is only applicable if the TOE is distributed and therefore has multiple components that need to communicate via an internal TSF channel. When creating the TSF from the initial pair of components, either of these components may be identified as the TSF for the purposes of satisfying the meaning of "TSF" in this SFR.

The intention of this requirement is to ensure that there is a registration process that includes a positive enablement step by an administrator before components joining a distributed TOE can communicate with the other components of the TOE and before the new component can act as part of the TSF. The registration process may itself involve communication with the joining component: many implementations use a bespoke process for this, and the security requirements for the "registration communication" are then defined in FCO_CPC_EXT.1.2. Use of this "registration communication" channel is not deemed inconsistent with the requirement of FCO_CPC_EXT.1.1 (i.e., the registration channel can be used before the enablement step, but only in order to complete the registration process).

The channel selection (for the registration channel) in FCO_CPC_EXT.1.2 is essentially a choice between the use of a normal secure channel that is equivalent to a channel used to communicate with external IT entities (FTP_ITC.1) or existing TOE components (FPT_ITT.1/INTER_XFER and FPT_ITT.1/INTER_XFER_AGENT), or else a separate type of channel that is specific to registration (FTP_TRP.1/TRUSTPATH_ENROLL or FTP_TRP.1/TRUSTPATH_JOIN). If the TOE does not require a communications

channel for registration (e.g., because the registration is achieved entirely by configuration actions by an administrator at each of the components) then the main selection in FCO_CPC_EXT.1.2 is completed with the "No channel" option.

If the ST author selects the FTP_ITC.1 or FPT_ITT.1/INTER_XFER and FPT_ITT.1/INTER_XFER_AGENT channel type in the main selection in FCO_CPC_EXT.1.2 then the TSS identifies the relevant SFR iteration that specifies the channel used. If the ST author selects the FTP_TRP.1/TRUSTPATH_ENROLL or FTP_TRP.1/TRUSTPATH_JOIN channel type, then the TSS (possibly with support from the operational guidance) describes details of the channel and the mechanisms that it uses (and describes how the registration process ensures that the channel can only be used by the intended joiner and gatekeeper). Note that the FTP_TRP.1/TRUSTPATH_ENROLL or FTP_TRP.1/TRUSTPATH_JOIN channel type may require support from security measures in the operational environment (see the definition of FTP_TRP.1/TRUSTPATH_ENROLL or FTP_TRP.1/TRUSTPATH_JOIN for details).

If the ST author selects the FTP_ITC.1 or FPT_ITT.1/INTER_XFER / FPT_ITT.1/INTER_XFER_AGENT channel type in the main selection in FCO_CPC_EXT.1.2 then the ST identifies the registration channel as a separate iteration of FTP_ITC.1 or FPT_ITT.1/INTER_XFER / FPT_ITT.1/INTER_XFER_AGENT and gives the iteration identifier (e.g., "FPT_ITT.1/Join") in an ST Application Note for FCO_CPC_EXT.1.

Note that the channel that is set up and used for registration may be adopted as a continuing internal communication channel (i.e., between different TOE components) provided that the channel meets the requirements of FTP_ITC.1 or FPT_ITT.1/INTER_XFER / FPT_ITT.1/INTER_XFER_AGENT. Otherwise the registration channel is closed after use and a separate channel is used for the internal communications.

Specific requirements for Preparative Procedures relating to FCO_CPC_EXT.1 are defined in the Evaluation Activities.

Evaluation Activities

FCO_CPC_EXT.1
TSS
If "invoke platform-provided functionality" is selected, the evaluator shall examine the TSS to verify that it describes (for each supported platform) how the audit record protection functionality is invoked (it should be noted that this may be through a mechanism that is not implemented by the MDM server; nonetheless, that mechanism will be identified in the TSS as part of this evaluation activity).

Guidance
The evaluator shall examine the guidance documentation to confirm that it contains instructions for enabling and disabling communications with any individual component of a distributed TOE. The evaluator shall confirm that the method of disabling is such that all other components can be prevented from communicating with the component that is being removed from the TOE (preventing the remaining components from either attempting to initiate communications to the disabled component, or from responding to communications from the disabled component).

Tests
- *Test FCO_CPC_EXT.1:1: The evaluator shall confirm that an IT entity that is not currently a member of the distributed TOE cannot communicate with any component of the TOE until the non-member entity is enabled by an administrator for each of the non-equivalent TOE components that it is required to communicate with (non-equivalent TOE components are as defined in the minimum configuration for the distributed TOE)*
- *Test FCO_CPC_EXT.1:2: The evaluator shall confirm that after enablement, an IT entity can communicate only with the components that it has been enabled for. This includes testing that the enabled communication is successful for the enabled component pair, and that communication remains unsuccessful with any other component for which communication is possible but has not been explicitly enabled.*

  Some TOEs may set up the registration channel before the enablement step is carried out, but in such a case the channel must not allow communications until after the enablement step has been completed.
- *Test FCO_CPC_EXT.1:3: The evaluator shall separately disable each TOE component in turn and ensure that the other TOE components cannot then communicate with the disabled component, whether by attempting to initiate communications with the disabled component or by responding to communication attempts from the disabled component. In situations where one component acts as the 'Gatekeeper' for all other components, the test would*

involve disabling the components in turn on the Gatekeeper and ensuring that the TOE no longer communicates with disabled components.

4.1.4 Cryptographic Support (FCS)

FCS_CKM.1 Cryptographic Key Generation

FCS_CKM.1.1

The TSF shall [**selection**: *invoke platform-provided functionality, implement functionality*] to generate **asymmetric** cryptographic keys in accordance with a specified cryptographic key generation algorithm [**selection**:

- *RSA schemes using a cryptographic key size of 3072-bits that meet the following: FIPS PUB 186-5, "Digital Signature Standard (DSS)," Appendix A.1*
- *ECC schemes using "NIST curve" P-384 that meet the following: FIPS PUB 186-5, "Digital Signature Standard (DSS)," Appendix A.2*
- *FFC schemes using "safe-prime" groups that meet the following: 'NIST Special Publication 800-56A Revision 3, "Recommendation for Pair-Wise Key Establishment Schemes Using Discrete Logarithm Cryptography"', and [**selection**: RFC 3526, RFC 7919]*

].

Application Note: The ST author must select all key generation schemes used for key establishment and MDM authentication. When key generation is used for key establishment, the schemes in FCS_CKM.2.1 and selected cryptographic protocols must match the selection. When key generation is used for MDM authentication, the public key is expected to be associated with an X.509v3 certificate.

If the TOE only acts as a receiver in the RSA key establishment scheme, the TOE does not need to implement RSA key generation.

In a distributed TOE, if the TOE component acts as a receiver in the key establishment scheme, the TOE does not need to implement key generation.

Evaluation Activities

FCS_CKM.1
TSS
If "invoke platform-provided functionality" is selected: The evaluator shall examine the TSS to verify that it describes (for each supported platform) how the key generation functionality is invoked (it should be noted that this may be through a mechanism that is not implemented by the MDM server; nonetheless, that mechanism will be identified in the TSS as part of this evaluation activity).

If "implement functionality" is selected: The evaluator shall ensure that the TSS identifies the key sizes supported by the TOE. If the ST specifies more than one scheme, the evaluator shall examine the TSS to verify that it identifies the usage for each scheme.

Guidance
The evaluator shall verify that the operational guidance instructs the administrator how to configure the TOE to use the selected key generation schemes and key sizes for all uses defined in this PP.

Tests
Key Generation for FIPS PUB 186-5 RSA Schemes

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

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

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

- *Primes p1, p2, q1, and q2 shall be provable primes and p and q shall be probable primes*
- *Primes p1, p2, q1, q2, p, and q shall all be probable primes*

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

If possible, the Random Probable primes method should also be verified against a known good implementation as described above. Otherwise, the evaluator shall have the TSF generate 10 keys pairs for each supported key length nlen and verify:

- *n = p*q,*
- *p and q are probably prime according to Miller-Rabin tests,*
- *GCD(p-1,e) = 1,*
- *GCD(q-1,e) = 1,*
- *2^16 <= e <= 2^256 and e is an odd integer,*
- *|p-q| > 2^(nlen/2 - 100),*
- *p >= squareroot(2)*(2^(nlen/2 -1)),*
- *q >= squareroot(2)*(2^(nlen/2 -1)),*
- *2^(nlen/2) < d < LCM(p-1,q-1),*
- *e*d = 1 mod LCM(p-1,q-1).*

Key Generation for Elliptic Curve Cryptography (ECC)

FIPS 186-5 ECC Key Generation Test

The evaluator shall require the implementation under test (IUT) to generate 10 private or public key pairs. The private key shall be generated using an approved random bit generator (RBG). To determine correctness, the evaluator shall submit the generated key pairs to the public key verification (PKV) function of a known good implementation.

FIPS 186-5 Public Key Verification (PKV) Test

The evaluator shall generate 10 private or public key pairs using the key generation function of a known good implementation and modify five of the public key values so that they are incorrect, leaving five values unchanged (i.e., correct). The evaluator shall obtain in response a set of 10 PASS or FAIL values.

Key Generation for Finite-Field Cryptography (FFC)

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

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

Cryptographic and Field Primes:

- *Primes q and p shall both be provable primes*
- *Primes q and field prime p shall both be probable primes and two ways to generate the cryptographic group generator g:*

Cryptographic Group Generator:

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

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

Private Key:

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

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

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

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

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

for each FFC parameter set and key pair.

FFC Schemes using safe-prime groups

Testing for FFC Schemes using safe-prime groups is done as part of testing in CKM.2.1.

FCS_CKM.2 Cryptographic Key Establishment

FCS_CKM.2.1

The TSF shall [**selection**: *invoke platform-provided functionality, implement functionality*] **to perform** cryptographic **key establishment** in accordance with a specified cryptographic key **establishment** method: [**selection**:

- *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"*
- *FFC schemes using "safe-prime" groups that meet the following: 'NIST Special Publication 800-56A Revision 3, "Recommendation for Pair-Wise Key Establishment Schemes Using Discrete Logarithm Cryptography"', and [**selection**: RFC 3526, RFC 7919]*

].

Application Note: The ST author must select all key establishment schemes used for the selected cryptographic protocols.

The elliptic curves used for the key establishment scheme must correlate with the curves specified in FCS_CKM.1.1.

Evaluation Activities

FCS_CKM.2
TSS
If "invoke platform-provided functionality" is selected:

The evaluator shall examine the TSS to verify that it describes (for each supported platform) how the key establishment functionality is invoked (it should be noted that this may be through a mechanism that is not implemented by the MDM server; nonetheless, that mechanism will be identified in the TSS as part of this evaluation activity).

If "implement functionality" is selected:

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

The evaluator shall ensure that the TSS describes how the TOE handles decryption errors. In accordance with NIST Special Publication 800-56B, the TOE must not reveal the particular error that occurred, either through the contents of any outputted or logged error message or through timing variations. If KTS-OAEP is supported, the evaluator shall create separate contrived ciphertext values that trigger each of the three decryption error checks described in NIST Special Publication 800-56B section 7.2.2.3, ensure that each decryption attempt results in an error, and ensure that any outputted or logged error message is identical for each. If KTS-KEM-KWS is supported, the evaluator shall create separate contrived ciphertext values that trigger each of the three decryption error checks described in NIST Special Publication 800-56B section 7.2.3.3, ensure that each decryption attempt results in an error, and ensure that any outputted or logged error message is identical for each.

Guidance

The evaluator shall verify that the operational guidance instructs the administrator how to configure the TOE to use the selected key establishment schemes.

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

SP800-56A Key Establishment Schemes

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

Function Test

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

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

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

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

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

Validity Test

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

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

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

RSA-based key establishment

The evaluator shall verify the correctness of the TSF's implementation of RSAES-PKCS1-v1_5 by using a known good implementation for each protocol selected in FTP_TRP.1/TRUSTPATH_REM_ADMIN, FTP_TRP.1/TRUSTPATH_ENROLL,

FTP_TRP.1/TRUSTPATH_JOIN, *FTP_ITC.1/INTER_XFER_IT*, *FTP_ITC.1/INTER_TSF_XFER_AGENT*, *FPT_ITT.1/INTER_XFER*, *and FPT_ITT.1/INTER_XFER_AGENT that uses RSAES-PKCS1-v1_5.*

FFC Schemes using safe-prime groups

The evaluator shall verify the correctness of the TSF's implementation of safe-prime groups by using a known good implementation for each protocol selected in FTP_TRP.1/TRUSTPATH_REM_ADMIN, FTP_TRP.1/TRUSTPATH_ENROLL, FTP_TRP.1/TRUSTPATH_JOIN, FTP_ITC.1/INTER_XFER_IT, FTP_ITC.1/INTER_TSF_XFER_AGENT, FPT_ITT.1/INTER_XFER, and FPT_ITT.1/INTER_XFER_AGENT that uses safe-prime groups. This test must be performed for each safe-prime group that each protocol uses.

FCS_CKM.6 Timing and Event of Cryptographic Key Destruction

FCS_CKM.6.1

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

Application Note: Key destruction procedures are performed in accordance with FCS_CKM.6.2. Even if "invoking platform-provided functionality with the following rules" is selected in FCS_CKM.6.2, the TSF must determine when the plaintext keying material and CSP are no longer needed and thus should be destroyed. The TSF must release the key material and CSP when no longer needed, regardless if the TSF or TOE platform destroys the key material and CSPs.

For the purposes of this requirement, plaintext keying material refers to authentication data, authorization data, secret or private symmetric keys, data used to derive keys, etc.

FCS_CKM.6.2

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

- *invoking platform-provided functionality with the following rules:*
  - *For volatile memory, the destruction shall be executed by [**selection**:*
    - *a single direct overwrite consisting of [**selection**: a pseudo-random pattern using the TSF or platform RBG (as specified in FCS_RBG.1), zeroes, ones, a new value of a key, [**assignment**: some value that does not contain any CSP]]*
    - *removal of power to the memory*
    - *destruction of reference to the key directly followed by a request for garbage collection*

    ]
  - *For non-volatile memory that consists of the invocation of an interface provided by the underlying platform that [**selection**:*
    - *logically addresses the storage location of the key and performs a [**selection**: single, [**assignment**: ST author defined multi-pass]] direct overwrite consisting of [**selection**: a pseudo-random pattern using the TSF or platform RBG (as specified in FCS_RBG.1), zeroes, ones, a new value of a key, [**assignment**: some value that does not contain any CSP]]*
    - *instructs the underlying platform to destroy the abstraction that represents the key*

    ]
- *implementing key destruction in accordance with the following rules:*
  - *For volatile memory, the destruction shall be executed by a single direct overwrite [**selection**: consisting of a pseudo-random pattern using the TSF or platform RBG (as specified in FCS_RBG.1), 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 or platform RBG (as specified in FCS_RBG.1), followed by a read-verify.*
  - *For non-volatile flash memory, that is not wear-leveled, the destruction shall be executed [**selection**: by a single direct overwrite consisting of zeros followed by a read-verify, by a block erase that erases the reference to memory that stores data as well as the data itself]*
  - *For non-volatile flash memory, that is wear-leveled, the destruction*

*shall be executed [**selection**: by a single direct overwrite consisting of zeros, by a block erase]*
- *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*

].

Application Note: The ST author should select "invoking platform-provided functionality" if the MDM server performs no operations using plaintext secret, private cryptographic keys, and CSPs.

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

The zeroization indicated above applies to each intermediate storage area for plaintext key and Cryptographic Service Provider (CSP) (i.e., any storage, such as memory buffers, that is included in the path of such data) upon the transfer of the key or CSP to another location.

Since the TOE does not include the host IT environment, the extent of this capability is necessarily somewhat limited. For the purposes of this requirement, it is sufficient for the TOE to invoke the correct underlying functions of the host to perform the zeroization--it does not imply that the TOE has to include a kernel-mode memory driver to ensure the data are zeroized. It is assumed that the host platform appropriately performs zeroization of key material in its internal processes.

Several selections allow assignment of a "some value that does not contain any CSP." This means that the TOE uses some other specified data not drawn from a source that may contain key material or reveal information about key material, and not being any of the particular values listed as other selection options. The point of the phrase "does not contain any CSP" is to ensure that the overwritten data is carefully selected, and not taken from a general 'pool' that might contain current or residual data that itself requires confidentiality protection.

Evaluation Activities

FCS_CKM.6
Evaluation Activity Note: The evaluation activity used is dependent on the selection made in FCS_CKM.6.2.

TSS
The evaluator shall check to ensure the TSS lists each type of plaintext key material and CSP (authentication data, authorization data, secret or private symmetric keys, data used to derive keys, etc.) and its origin and storage location.

The evaluator shall verify that the TSS describes when each type of key material and CSP is no longer needed.

If "invoking platform-provided functionality with the following rules" is selected:

The evaluator shall examine the TSS to verify that it describes (for each supported platform) how the key releasing functionality is invoked (it should be noted that this may be through a mechanism that is not implemented by the MDM server; nonetheless, that mechanism will be identified in the TSS as part of this evaluation activity).

If "implementing key destruction in accordance with the following rules:" is selected:

The evaluator shall also verify that, for each type, the type of clearing procedure that is performed is listed. If different types of memory are used to store the materials to be protected, the evaluator shall check to ensure that the TSS describes the clearing procedure in terms of the memory in which the data are stored (for example, "secret keys stored on flash are cleared by overwriting once with zeros, while secret keys stored on the internal persistent storage device are cleared by overwriting one time with a random pattern that is changed before each write"). For block erases, the evaluator shall also ensure that the block erase command used is listed and shall verify that the command used also addresses any copies of the plaintext key material that may be created in order to optimize the use of flash memory.

Guidance
TBD

Tests

For each software and firmware key clearing situation the evaluator shall repeat the following tests. Note that at this time hardware-bound keys are explicitly excluded from testing.

- *Test FCS_CKM.6:1: The evaluator shall use appropriate combinations of specialized operational environment and development tools (debuggers, simulators, etc.) for the TOE and instrumented TOE builds to test that keys are cleared correctly, including all intermediate copies of the key that may have been created internally by the TOE during normal cryptographic processing with that key.*

  Cryptographic TOE implementations in software shall be loaded and exercised under a debugger to perform such tests. The evaluator shall perform the following test for each key subject to clearing, including intermediate copies of keys that are persisted encrypted by the TOE:
  1. *Load the instrumented TOE build in a debugger.*
  2. *Record the value of the key in the TOE subject to clearing.*
  3. *Cause the TOE to perform a normal cryptographic processing with the key from #1.*
  4. *Cause the TOE to clear the key.*
  5. *Cause the TOE to stop the execution but not exit.*
  6. *Cause the TOE to dump the entire memory footprint of the TOE into a binary file.*
  7. *Search the content of the binary file created in #4 for instances of the known key value from #1.*

  The test succeeds if no copies of the key from #1 are found in step #7 above and fails otherwise. The evaluator shall perform this test on all keys, including those persisted in encrypted form, to ensure intermediate copies are cleared.

- *Test FCS_CKM.6:2: In cases where the TOE is implemented in firmware and operates in a limited operating environment that does not allow the use of debuggers, the evaluator shall use a simulator for the TOE on a general purpose operating system. The evaluator shall provide a rationale explaining the instrumentation of the simulated test environment and justifying the obtained test results.*

FCS_COP.1/CONF_ALG Cryptographic Operation (Confidentiality Algorithms)

FCS_COP.1.1/CONF_ALG

The TSF shall [**selection**: *invoke platform-provided functionality, implement functionality*] **to** perform [*encryption and decryption*] in accordance with a specified cryptographic algorithm: [**selection**:

- *AES-CBC (as defined in FIPS PUB 197, and NIST SP 800-38A) mode*
- *AES-GCM (as defined in NIST SP 800-38D)*
- *AES Key Wrap (KW) (as defined in NIST SP 800-38F)*
- *AES Key Wrap with Padding (KWP) (as defined in NIST SP 800-38F)*
- *AES-CCM (as defined in NIST SP 800-38C)*

] and a cryptographic key size of 256-bits.

Application Note: For the second selection of FCS_COP.1.1/CONF_ALG, the ST author should choose the mode or modes in which AES operates in the trusted channel protocols. For the third selection, the ST author should choose the key sizes that are supported by this functionality.

Evaluation Activities

FCS_COP.1/CONF_ALG
TSS
If "invoke platform-provided functionality" is selected:

The evaluator shall examine the TSS to verify that it describes (for each supported platform) how the encryption and decryption functionality is invoked for each mode and key size selected in the MDM server's ST (it should be noted that this may be through a mechanism that is not implemented by the MDM server; nonetheless, that mechanism will be identified in the TSS as part of this evaluation activity).

Guidance
TBD

Tests
If "implement functionality" is selected:

AES-CBC Tests

- *Test FCS_COP.1/CONF_ALG:1: AES-CBC Known Answer Tests*

  There are four Known Answer Tests (KATs), described below. In all KATs, the plaintext, ciphertext, and IV values shall be 256-bit blocks. The results from each test may either be obtained by the evaluator directly or by supplying the inputs to the implementer and receiving the results in response. To determine correctness, the evaluator shall compare the resulting values to those obtained by submitting the same inputs to a known good implementation.

  - *Test FCS_COP.1/CONF_ALG:1.1: KAT-1. To test the encrypt functionality of AES-CBC, the evaluator shall supply a set of five plaintext values and obtain the ciphertext value that results from AES-CBC encryption of the given plaintext using a key value of all zeros and an IV of all zeros. All five plaintext values shall be encrypted with a 256-bit all-zeros key.*

    To test the decrypt functionality of AES-CBC, the evaluator shall perform the same test as for encrypt, using five ciphertext values as input and AES-CBC decryption.
  - *Test FCS_COP.1/CONF_ALG:1.2: KAT-2. To test the encrypt functionality of AES-CBC, the evaluator shall supply a set of five key values and obtain the ciphertext value that results from AES-CBC encryption of an all-zeros plaintext using the given key value and an IV of all zeros. All five of the keys shall be 256-bit keys.*

    To test the decrypt functionality of AES-CBC, the evaluator shall perform the same test as for encrypt, using an all-zero ciphertext value as input and AES-CBC decryption.
  - *Test FCS_COP.1/CONF_ALG:1.3: KAT-3. To test the encrypt functionality of AES-CBC, the evaluator shall supply the set of key values described below and obtain the ciphertext value that results from AES encryption of an all-zeros plaintext using the given key value and an IV of all zeros. The set shall have 256 256-bit keys. Key i in each set shall have the leftmost i bits be ones and the rightmost N-i bits be zeros, for i in [1,N].*

    To test the decrypt functionality of AES-CBC, the evaluator shall supply the set of key and ciphertext value pairs described below and obtain the plaintext value that results from AES-CBC decryption of the given ciphertext using the given key and an IV of all zeros. The set of key and ciphertext pairs shall have 256 256-bit key and ciphertext pairs. Key i in each set shall have the leftmost i bits be ones and the rightmost N-i bits be zeros, for i in [1,N]. The ciphertext value in each pair shall be the value that results in an all-zeros plaintext when decrypted with its corresponding key.
  - *Test FCS_COP.1/CONF_ALG:1.4: KAT-4. To test the encrypt functionality of AES-CBC, the evaluator shall supply the set of 256 plaintext values described below and obtain the two ciphertext values that result from AES-CBC encryption of the given plaintext using a 256-bit key value of all zeros with an IV of all zeros and using a 256-bit key value of all zeros with an IV of all zeros, respectively. Plaintext value i in each set shall have the leftmost i bits be ones and the rightmost 256-i bits be zeros, for i in [1,256].*

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

- *Test FCS_COP.1/CONF_ALG:2: AES-CBC Multi-Block Message Test*

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

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

- *Test FCS_COP.1/CONF_ALG:3: AES-CBC Monte Carlo Tests*

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

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

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

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

AES-GCM Test

- *Test FCS_COP.1/CONF_ALG:4: The evaluator shall test the authenticated encrypt functionality of AES-GCM for each combination of the following input parameter lengths:*
    - *256-bit keys*
    - *Two plaintext lengths. One of the plaintext lengths shall be a non-zero integer multiple of 128 bits, if supported. The other plaintext length shall not be an integer multiple of 128 bits, if supported.*
    - *Three AAD lengths. One AAD length shall be 0, if supported. One AAD length shall be a non-zero integer multiple of 128 bits, if supported. One AAD length shall not be an integer multiple of 128 bits, if supported.*
    - *Two IV lengths. If 96 bit IV is supported, 96 bits shall be one of the two IV lengths tested.*

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

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

The results from each test may either be obtained by the evaluator directly or by supplying the inputs to the implementer and receiving the results in response. To determine correctness, the evaluator shall compare the resulting values to those obtained by submitting the same inputs to a known good implementation.

AES-CCM Tests

- *Test FCS_COP.1/CONF_ALG:5: The evaluator shall test the generation-encryption and decryption-verification functionality of AES-CCM for the following input parameter and tag lengths:*
    - *256-bit keys*
    - *Two payload lengths. One payload length shall be the shortest supported payload length, greater than or equal to zero bytes. The other payload length shall be the longest supported payload length, less than or equal to 32 bytes (256 bits).*
    - *Two or three associated data lengths. One associated data length shall be 0, if supported. One associated data length shall be the shortest supported payload length, greater than or equal to zero bytes. One associated data length shall be the longest supported payload length, less than or equal to 32 bytes (256 bits). If the implementation supports an associated data length of 216 bytes, an associated data length of 216 bytes shall be tested.*
    - *Nonce lengths. All supported nonce lengths between 7 and 13 bytes, inclusive, shall be tested.*
    - *Tag lengths. All supported tag lengths of 4, 6, 8, 10, 12, 14 and 16 bytes shall be tested.*

  To test the generation-encryption functionality of AES-CCM, the evaluator shall perform the following four tests:
    - *Test FCS_COP.1/CONF_ALG:5.1: For EACH supported key and associated data length and ANY supported payload, nonce and tag length, the evaluator shall supply one key value, one nonce value and 10 pairs of associated data and payload values and obtain the resulting ciphertext.*
    - *Test FCS_COP.1/CONF_ALG:5.2: For EACH supported key and payload length and ANY supported associated data, nonce and tag length, the evaluator shall supply one key value, one nonce value and 10 pairs of associated data and payload values and obtain the resulting ciphertext.*
    - *Test FCS_COP.1/CONF_ALG:5.3: For EACH supported key and nonce length and ANY supported associated data, payload and tag length, the evaluator shall supply one key value and 10 associated data, payload and nonce value 3-tuples and obtain the resulting ciphertext.*
    - *Test FCS_COP.1/CONF_ALG:5.4: . For EACH supported key and tag length and ANY supported associated data, payload and nonce length, the evaluator shall supply one key value, one nonce value and 10 pairs of associated data and payload values and obtain the resulting ciphertext.*

  To determine correctness in each of the above tests, the evaluator shall compare the ciphertext with the result of generation-encryption of the same inputs with a known good implementation.

  To test the decryption-verification functionality of AES-CCM, for EACH combination of supported associated data length, payload length, nonce length and tag length, the evaluator shall supply a key value and 15 nonce, associated data and ciphertext 3-tuples and obtain either a FAIL result or a PASS result with the decrypted payload. The evaluator shall supply 10 tuples that should FAIL and 5 that should PASS per set of 15.

  AES Key Wrap (AES-KW) and Key Wrap with Padding (AES-KWP) Test

- *Test FCS_COP.1/CONF_ALG:6: The evaluator shall test the authenticated encryption functionality of AES-KW for EACH combination of the following input parameter lengths:*
    - *256-bit key encryption keys (KEKs)*
    - *Three plaintext lengths. One of the plaintext lengths shall be two semi-blocks (128 bits). One of the plaintext lengths shall be three semi-blocks (192 bits). The third data unit length shall be the longest supported plaintext length less than or equal to 64*

semi-blocks (4096 bits).

using a set of 100 key and plaintext pairs and obtain the ciphertext that results from AES-KW authenticated-encryption. To determine correctness, the evaluator shall use the AES-KW authenticated-encryption function of a known good implementation.

The evaluator shall test the authenticated-decryption functionality of AES-KW using the same test as for authenticated-encryption, replacing plaintext values with ciphertext values and AES-KW authenticated-encryption with AES-KW authenticated-decryption.

The evaluator shall test the authenticated-encryption functionality of AES-KWP using the same test as for AES-KW authenticated-encryption with the following change in the three plaintext lengths:

- *One plaintext length shall be one octet.*
- *One plaintext length shall be 20 octets (160-bits).*
- *One plaintext length shall be the longest supported plaintext length less than or equal to 512 octets (4096-bits).*

The evaluator shall test the authenticated-decryption functionality of AES-KWP using the same test as for AES-KWP authenticated-encryption, replacing plaintext values with ciphertext values and AES-KWP authenticated-encryption with AES-KWP authenticated-decryption.

FCS_COP.1/HASH_ALG Cryptographic Operation (Hashing Algorithms)

FCS_COP.1.1/HASH_ALG

The TSF shall [**selection**: *invoke platform-provided functionality, implement functionality*] **to** perform **cryptographic hashing** in accordance with specified cryptographic algorithm [*SHA-384*] **and message digest size** [*384-bits*] that meet the following: [*FIPS Pub 180-4*.]

Application Note: The intent of this requirement is to specify the hashing function for trusted channel protocols. The hash selection must support the message digest size selection. The hash selection should be consistent with the overall strength of the algorithm used.

Evaluation Activities

FCS_COP.1/HASH_ALG
TSS
If "invoke platform-provided functionality" is selected:

The evaluator shall examine the TSS to verify that it describes (for each supported platform) how the hash functionality is invoked for each digest size selected in the MDM server's ST (it should be noted that this may be through a mechanism that is not implemented by the MDM server; nonetheless, that mechanism will be identified in the TSS as part of this evaluation activity).

If "implement functionality" is selected:

The evaluator shall check that the association of the hash function with other TSF cryptographic functions (for example, the digital signature verification function) is documented in the TSS. The evaluator checks the AGD documents to determine that any configuration that is required to be done to configure the functionality for the required hash sizes is present. The TSF hashing functions can be implemented in one of two modes. The first mode is the byte-oriented mode. In this mode the TSF only hashes messages that are an integral number of bytes in length; i.e., the length (in bits) of the message to be hashed is divisible by 8. The second mode is the bit-oriented mode. In this mode the TSF hashes messages of arbitrary length. As there are different tests for each mode, an indication is given in the following sections for the bit-oriented vs. byte-oriented TestMAC.

Guidance
TBD

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

Short Messages Test Bit-oriented Mode

- *Test FCS_COP.1/HASH_ALG:1: The evaluator devises an input set consisting of m+1 messages, where m is the block length of the hash algorithm. The length of the messages range sequentially from 0 to m bits. The message text shall be pseudorandomly generated. The evaluator computes the message digest for each of the messages and ensure that the correct result is produced when the messages are provided to the TSF.*

  Short Messages Test Byte-oriented Mode

- *Test FCS_COP.1/HASH_ALG:2: The evaluator devises an input set consisting of m/8+1 messages, where m is the block length of the hash algorithm. The length of the messages range sequentially from 0 to m/8 bytes, with each message being an integral number of bytes. The message text shall be pseudorandomly generated. The evaluator computes the message digest for each of the messages and ensure that the correct result is produced when the messages are provided to the TSF.*

  Selected Long Messages Test Bit-oriented Mode

- *Test FCS_COP.1/HASH_ALG:3: The evaluator devises an input set consisting of m messages, where m is the block length of the hash algorithm. The length of the ith message is 512 + 99*i, where 1 ≤ i ≤ m. The message text shall be pseudorandomly generated. The evaluator computes the message digest for each of the messages and ensure that the correct result is produced when the messages are provided to the TSF.*

  Selected Long Messages Test Byte-oriented Mode

- *Test FCS_COP.1/HASH_ALG:4: The evaluator devises an input set consisting of m/8 messages, where m is the block length of the hash algorithm. The length of the ith message is 512 + 8*99*i, where 1 ≤ i ≤ m/8. The message text shall be pseudorandomly generated. The evaluator computes the message digest for each of the messages and ensure that the correct result is produced when the messages are provided to the TSF.*

  Pseudorandomly Generated Messages Test

- *Test FCS_COP.1/HASH_ALG:5: This test is for byte-oriented implementations only. The evaluator randomly generates a seed that is n bits long, where n is the length of the message digest produced by the hash function to be tested. The evaluator then formulates a set of 100 messages and associated digests by following the algorithm provided in Figure 1 of [SHAVS]. The evaluator then ensures that the correct result is produced when the messages are provided to the TSF.*

FCS_COP.1/KEY_HASH Cryptographic Operation (Keyed-Hash Message Authentication)

FCS_COP.1.1/KEY_HASH

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

Application Note: The intent of this requirement is to specify the keyed-hash message authentication function used when used for key establishment purposes for the various cryptographic protocols used by the TOE (e.g., trusted channel). The hash selection must support the message digest size selection. The hash selection should be consistent with the overall strength of the algorithm used for FCS_COP.1/SIGN_ALG.

Evaluation Activities

FCS_COP.1/KEY_HASH
TSS
If "invoke platform-provided functionality" is selected:

The evaluator shall examine the TSS to verify that it describes (for each supported platform) how the keyed-hash functionality is invoked for each mode and key size selected in the MDM server's ST (it should be noted that this may be through a mechanism that is not implemented by the MDM server; nonetheless, that mechanism will be identified in the TSS as part of this evaluation activity).

If "implement functionality" is selected:

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

Guidance
TBD

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

FCS_COP.1/SIGN_ALG Cryptographic Operation (Signature Algorithms)

FCS_COP.1.1/SIGN_ALG

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

- *RSA schemes using a cryptographic key size of 3072-bits that meet the following: FIPS PUB 186-5, "Digital Signature Standard (DSS)," Section 4*
- *ECDSA schemes using "NIST curve" P-384 that meet the following: FIPS PUB 186-5, "Digital Signature Standard (DSS)," Section 5*

]

Application Note: The ST Author should choose the algorithm implemented to perform digital signatures. The MDM server must perform digital signatures in accordance with the trusted channel protocols. The MDM server is required to validate any signed policies and policy updates sent by the MDM server.

Evaluation Activities

FCS_COP.1/SIGN_ALG
TSS
If "invoke platform-provided functionality" is selected:

The evaluator shall examine the TSS to verify that it describes (for each supported platform) how the digital signature functionality is invoked for each operation they are used for in the MDM server (it should be noted that this may be through a mechanism that is not implemented by the MDM server; nonetheless, that mechanism will be identified in the TSS as part of this evaluation activity).

Guidance
TBD

Tests
If "implement functionality" is selected:

ECDSA Algorithm Tests

- *Test FCS_COP.1/SIGN_ALG:1: ECDSA FIPS 186-5 Signature Generation Test*

For the supported NIST curve and function pair, the evaluator shall generate 10 1024-bit long messages and obtain for each message a public key and the resulting signature values

R and S. To determine correctness, the evaluator shall use the signature verification function of a known good implementation.

- *Test FCS_COP.1/SIGN_ALG:2: ECDSA FIPS 186-5 Signature Verification Test*

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

  RSA Signature Algorithm Tests

- *Test FCS_COP.1/SIGN_ALG:3: Signature Generation Test*

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

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

- *Test FCS_COP.1/SIGN_ALG:4: Signature Verification Test*

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

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

FCS_HTTPS_EXT.1 HTTPS Protocol

This is a selection-based component. Its inclusion depends upon selection from FPT_ITT.1.1/INTER_XFER, FPT_ITT.1.1/INTER_XFER_AGENT, FTP_ITC.1.1/INTER_TSF_XFER_AGENT, FTP_ITC.1.1/INTER_XFER_IT, FTP_TRP.1.1/TRUSTPATH_ENROLL, FTP_TRP.1.1/TRUSTPATH_REM_ADMIN.

FCS_HTTPS_EXT.1.1

The TSF shall implement the HTTPS 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.

Application Note: The TLS Functional Package must be included in the ST, with the following selections made:

- FCS_TLS_EXT.1:
  - TLS must be selected
  - Either client or server is selected as appropriate
- FCS_TLSC_EXT.1.1 or FCS_TLSS_EXT.1.1 (as appropriate):
  - The cipher suites selected must correspond with the algorithms and hash functions allowed in FCS_COP.1.

Validity is determined by the certificate path, the expiration date, and the revocation status in accordance with RFC 5280.

The requirement is claimed if the TSF selects HTTPS in any iteration of FPT_ITT.1, FTP_ITC.1, or FTP_TRP.1.

Evaluation Activities

FCS_HTTPS_EXT.1
TSS
TBD

Guidance
TBD

Tests
- *Test FCS_HTTPS_EXT.1:1: The evaluator shall attempt to establish an HTTPS connection with a web server, observe the traffic with a packet analyzer, and verify that the connection succeeds and that the traffic is identified as TLS or HTTPS.*

Other tests are performed in conjunction with the TLS evaluation activities.

FCS_IV_EXT.1 Initialization Vector Generation

This is a selection-based component. Its inclusion depends upon selection from FCS_STG_EXT.1.1.

FCS_IV_EXT.1.1

The TSF shall [**selection**: *invoke platform-provided functionality*, *implement functionality*] to generate IVs in accordance with Table 4.

Application Note: This requirement must be included in the ST if the selection in FCS_STG_EXT.1 indicates that the TSF is protecting private keys and persistent secrets with encryption rather than the platform-provided key storage.

Table 4 lists the requirements for composition of IVs according to the corresponding NIST Special Publications for each cipher mode. The composition of IVs generated for encryption according to a cryptographic protocol is addressed by the protocol. Thus, this requirement addresses only IVs generated for key storage encryption.

Table 4: References and IV Requirements for NIST-approved Cipher Modes

Cipher Mode	Reference	IV Requirement
Electronic Codebook (ECB)	SP800-38A	No IV
Counter (CTR)	SP800-38A	"Initial Counter" shall be non-repeating. No counter value shall be repeated across multiple messages with the same secret key.
Cipher Block Chaining (CBC)	SP800-38A	IVs shall be unpredictable. Repeating IVs leak information about whether the first one or more blocks are shared between two messages, so IVs should be non-repeating in such situations.
Output Feedback (OFB)	SP800-38A	IVs shall be non-repeating and shall not be generated by invoking the cipher on another IV.
Cipher Feedback (CFB)	SP800-38A	IVs should be non-repeating as repeating IVs leak information about the first plaintext block and about common shared

		prefixes in messages.
XOR Encrypt XOR (XEX) Tweakable Block Cipher with Ciphertext Stealing (XTS)	SP800-38E	No IV. Tweak values shall be non-negative integers, assigned consecutively, and starting at an arbitrary non-negative integer.
Cipher-based Message Authentication Code (CMAC)	SP800-38B	No IV
Key Wrap and Key Wrap with Padding	SP800-38F	No IV
Counter with CBC-Message Authentication Code (CCM)	SP800-38C	No IV. Nonces shall be non-repeating.
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).

Evaluation Activities

FCS_IV_EXT.1
TSS
If "invoke platform-provided functionality" is selected:

The evaluator shall examine the TSS to verify that it describes (for each supported platform) how the IV generation is invoked for each mode selected in the MDM server's ST (it should be noted that this may be through a mechanism that is not implemented by the MDM server; nonetheless, that mechanism will be identified in the TSS as part of this evaluation activity).

If "implement functionality" is selected:

The evaluator shall examine the TSS to ensure that it details the encryption of user credentials, persistent secrets, and private keys and the generation of the IVs used for that encryption.

Guidance
TBD

Tests
The evaluator shall ensure that the generation of IVs for each key encrypted by the same KEK meets Table 4.

FCS_RBG.1 Random Bit Generation (RBG)

FCS_RBG.1.1

The TSF shall **[selection: *invoke platform-provided functionality, implement functionality*] to** perform **all** deterministic random bit generation services in accordance with NIST Special Publication 800-90A using [**selection**:

- *Hash_DRBG (any)*
- *HMAC_DRBG (any)*
- *CTR_DRBG (AES)*

] in accordance with [*NIST SP 800-90A*] after initialization with a seed.

Application Note: NIST SP 800-90A contains three different methods of generating random numbers; each of these, in turn, depends on underlying cryptographic primitives (hash functions/ciphers). The ST author will select the function used and include the specific underlying cryptographic primitives used in the requirement or in the TSS.

FCS_RBG.1.2

The TSF shall use a [**selection**: *TSF noise source [**assignment**: name of noise source], **multiple TSF noise sources [assignment: names of noise sources]** , TSF interface for seeding*] for initialized seeding.

Application Note: For the selection in this requirement, the ST author selects "TSF noise source" if a single noise source is used as input to the DRBG. The ST author selects "multiple TSF noise sources" if a seed is formed from a combination of two or more noise sources within the TOE boundary. If the TSF implements two or more separate DRBGs that are seeded in separate manners, this SFR should be iterated for each DRBG. It multiple distinct noise sources exist such that each DRBG only uses one of them, then each iteration would select "TSF noise source"; "multiple TSF noise sources" is only selected if a single DRBG uses multiple noise sources for its seed. The ST author selects "TSF interface for seeding" if noise source data is generated outside the TOE boundary.

If "TSF noise source" is selected, FCS_RBG.3 must be claimed.

If "multiple TSF noise sources" is selected, FCS_RBG.4 and FCS_RBG.5 must be claimed.

If "TSF interface for seeding" is selected, FCS_RBG.2 must be claimed.

FCS_RBG.1.3

The TSF shall update the RBG state by [**selection**: *reseeding, uninstantiating and reinstantiating*] using a [**selection**: *TSF noise source [**assignment**: name of noise source], TSF interface for seeding*] in the following situations: [**selection**:

- *never*
- *on demand*
- *on the condition: [**assignment**: condition]*
- *after [**assignment**: time]*

] in accordance with [**assignment**: *list of standards*].

Evaluation Activities

FCS_RBG.1.1
TSS
The evaluator shall verify that the TSS identifies the DRBGs used by the TOE.

If "invoke platform-provided functionality" is selected:

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

Guidance
If the DRBG functionality is configurable, the evaluator shall verify that the operational guidance includes instructions on how to configure this behavior.

Tests
If "implement functionality" is selected:

The evaluator shall perform the following tests.

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

If the RNG has prediction resistance enabled, each trial consists of (1) instantiate DRBG, (2) generate the first block of random bits, (3) generate a second block of random bits, and (4) uninstantiate. The evaluator verifies that the second block of random bits is the expected value. The evaluator shall generate eight input values for each trial. The first is a count (0 - 14). The next three are entropy input, nonce, and personalization string for the instantiate operation. The next two are additional input and entropy input for the first call to generate. The final two are additional input and entropy input for the second call to generate. These values are randomly generated. "generate one block of random bits" means to generate random bits with number of returned bits equal to the Output Block Length (as defined in NIST SP 800-90A).

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

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

Entropy input: the length of the entropy input value must equal the seed length.

Nonce: If a nonce is supported (CTR_DRBG with no Derivation Function does not use a nonce), the nonce bit length is one-half the seed length.

Personalization string: The length of the personalization string must be less than or equal to seed length. If the implementation only supports one personalization string length, then the same length can be used for both values. If more than one string length is support, the evaluator shall use personalization strings of two different lengths. If the implementation does not use a personalization string, no value needs to be supplied.

Additional input: the additional input bit lengths have the same defaults and restrictions as the personalization string lengths.

The evaluator shall perform the following tests:

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

If the RNG has prediction resistance enabled, each trial consists of (1) instantiate DRBG, (2) generate the first block of random bits (3) generate a second block of random bits (4) uninstantiate. The evaluator verifies that the second block of random bits is the expected value. The evaluator shall generate eight input values for each trial. The first is a count (0 – 14). The next three are entropy input, nonce, and personalization string for the instantiate operation. The next two are additional input and entropy input for the first call to generate. The final two are additional input and entropy input for the second call to generate. These values are randomly generated. "generate one block of random bits" means to generate random bits with number of returned bits equal to the Output Block Length (as defined in NIST SP 800-90A).

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

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

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

FCS_RBG.1.2
Documentation will be produced - and the evaluator shall perform the activities - in accordance with and the Clarification to the Entropy Documentation and Assessment Annex.
TSS
There are no additional TSS evaluation activities for this element.

Guidance
There are no additional Guidance evaluation activities for this element.

Tests
There are no test activities for this element.

FCS_RBG.1.3
TSS
The evaluator shall verify that the TSS identifies how the DRBG state is updated, and the situations under which this may occur.

Guidance
If the ST claims that the DRBG state can be updated on demand, the evaluator shall verify that

the operational guidance has instructions for how to perform this operation.

Tests
There are no test activities for this element.

FCS_RBG.2 Random Bit Generation (External Seeding)

This is a selection-based component. Its inclusion depends upon selection from FCS_RBG.1.2.

FCS_RBG.2.1

The TSF shall be able to accept a minimum input of [**assignment**: *minimum input length greater than zero*] from a TSF interface for the purpose of seeding.

Application Note: This requirement is claimed when a DRBG is seeded with entropy from one or more noise sources that is outside the TOE boundary. Typically the entropy produced by an environmental noise source is conditioned such that the input length has full entropy and is therefore usable as the seed. However, if this is not the case, it should be noted what the minimum entropy rate of the noise source is so that the TSF can collect a sufficiently large sample of noise data to be conditioned into a seed value.

This requirement is claimed if "TSF interface for seeding" is selected in FCS_RBG.1.2.

Evaluation Activities

FCS_RBG.2
The evaluator shall examine the entropy documentation required by FCS_RBG.1.2 to verify that it identifies, for each DRBG function implemented by the TOE, the TSF external interface used to seed the TOE's DRBG. The evaluator shall verify that this includes the amount of sampled data and the min-entropy rate of the sampled data such that it can be determined that sufficient entropy can be made available for the highest strength keys that the TSF can generate (e.g., 256 bits). If the seed data cannot be assumed to have full entropy (e.g., the min-entropy of the sampled bits is less than 1), the evaluator shall ensure that the entropy documentation describes the method by which the TOE estimates the amount of entropy that has been accumulated to ensure that sufficient data is collected and any conditioning that the TSF applies to the output data to create a seed of sufficient size with full entropy.

TSS
There are no additional TSS evaluation activities for this component.

Guidance
There are no additional Guidance evaluation activities for this component.

Tests
There are no test activities for this component.

FCS_RBG.3 Random Bit Generation (Internal Seeding - Single Source)

This is a selection-based component. Its inclusion depends upon selection from FCS_RBG.1.2.

FCS_RBG.3.1

The TSF shall be able to seed the RBG using a [**selection, choose one of**: *TSF software-based noise source, TSF hardware-based noise source [**assignment**: name of noise source]*] with a minimum of [**assignment**: *number of bits*] bits of min-entropy.

Application Note: This requirement is claimed when a DRBG is seeded with entropy from a single noise source that is within the TOE boundary. Min-entropy should be expressed as a ratio of entropy bits to sampled bits so that the total amount of data needed to ensure full entropy is known, as well as the conditioning function by which that data is reduced in size to the seed.

This requirement is claimed if "TSF noise source..." is selected in FCS_RBG.1.2.

Evaluation Activities

FCS_RBG.3
The evaluator shall examine the entropy documentation required by FCS_RBG.1.2 to verify that it identifies, for each DRBG function implemented by the TOE, the TSF noise source used to seed the TOE's DRBG. The evaluator shall verify that this includes the amount of sampled data and the min-entropy rate of the sampled data such that it can be determined that sufficient entropy can be made available for the highest strength keys that the TSF can generate (e.g., 256 bits). If the seed data cannot be assumed to have full entropy (e.g., the min-entropy of the sampled bits is less than 1), the evaluator shall ensure that the entropy documentation describes the method by which the TOE estimates the amount of entropy that has been accumulated to ensure that sufficient data is collected and any conditioning that the TSF applies to the output data to create a seed of sufficient size with full entropy.

TSS
There are no additional TSS evaluation activities for this component.

Guidance
There are no additional Guidance evaluation activities for this component.

Tests
There are no test activities for this component.

FCS_RBG.4 Random Bit Generation (Internal Seeding - Multiple Sources)

This is a selection-based component. Its inclusion depends upon selection from FCS_RBG.1.2.

FCS_RBG.4.1

The TSF shall be able to seed the RBG using [**selection**: *[**assignment**: number] TSF software-based noise sources, [**assignment**: number] TSF hardware-based noise sources*].

Application Note: This requirement is claimed when a DRBG is seeded with entropy from multiple noise sources that are within the TOE boundary. FCS_RBG.5 defines the mechanism by which these sources are combined to ensure sufficient minimum entropy.

This requirement is claimed if "multiple TSF noise sources..." is selected in FCS_RBG.1.2.

Evaluation Activities

FCS_RBG.4
The evaluator shall examine the entropy documentation required by FCS_RBG.1.2 to verify that it identifies, for each DRBG function implemented by the TOE, each TSF noise source used to seed the TOE's DRBG. The evaluator shall verify that this includes the amount of sampled data and the min-entropy rate of the sampled data from each data source.

TSS
There are no additional TSS evaluation activities for this component.

Guidance
There are no additional Guidance evaluation activities for this component.

Tests
There are no test activities for this component.

FCS_RBG.5 Random Bit Generation (Combining Noise Sources)

This is a selection-based component. Its inclusion depends upon selection from FCS_RBG.1.2.

FCS_RBG.5.1

The TSF shall [**assignment**: *combining operation*] [**selection**: *output from TSF noise sources, input from TSF interfaces for seeding*] to create the entropy input into the derivation function as defined in [**assignment**: *list of standards*], resulting in a minimum of [**assignment**: *number of bits*] bits of min-entropy.

Application Note: This requirement is claimed if "multiple TSF noise sources..." is selected in FCS_RBG.1.2.

Evaluation Activities

FCS_RBG.5
Using the entropy sources specified in FCS_RBG.4, the evaluator shall examine the entropy documentation required by FCS_RBG.1.2 to verify that it describes the method by which the various entropy sources are combined into a single seed. This should include an estimation of the rate at which each noise source outputs data and whether this is dependent on any system-specific factors so that each source's relative contribution to the overall entropy is understood. The evaluator shall verify that the resulting combination of sampled data and the min-entropy rate of the sampled data is described in sufficient detail to determine that sufficient entropy can be made available for the highest strength keys that the TSF can generate (e.g., 256 bits). If the seed data cannot be assumed to have full entropy (e.g., the min-entropy of the sampled bits is less than 1), the evaluator shall ensure that the entropy documentation describes the method by which the TOE estimates the amount of entropy that has been accumulated to ensure that sufficient data is collected and any conditioning that the TSF applies to the output data to create a seed of sufficient size with full entropy.

TSS
There are no additional TSS evaluation activities for this component.

Guidance
There are no additional Guidance evaluation activities for this component.

Tests
There are no test activities for this component.

FCS_STG_EXT.1 Cryptographic Key Storage

FCS_STG_EXT.1.1

The TSF shall use [**selection**: *platform-provided key storage, encryption as specified in FCS_STG_EXT.2*] for all persistent secrets and private keys.

Application Note: This requirement ensures that persistent secrets (credentials, secret keys) and private keys are stored securely when not in use. If some secrets or keys are manipulated by the TOE and others are manipulated by the platform, then both of the selections can be specified by the ST author and the ST author must identify in the TSS those keys which are manipulated by the TOE and those by the platform.

If "encryption as specified in FCS_STG_EXT.2" is selected then FCS_STG_EXT.2 and FCS_IV_EXT.1 must be included in the ST.

If the TSF is an application, and not a dedicated server, then it should store its private keys in the platform-provided key storage.

The ST author is responsible for selecting the manner in which the keys are stored and where they are stored in the selections above.

Evaluation Activities

FCS_STG_EXT.1
TSS
Regardless of whether this requirement is met by the TSF or the TOE platform, the evaluator will check the TSS to ensure that it lists each persistent secret (credential, secret key) and private key needed to meet the requirements in the ST. For each of these items, the evaluator will confirm that the TSS lists for what purpose it is used, and how it is stored. The evaluator then performs the following actions.

Persistent secrets and private keys manipulated by the TOE platform:

The evaluator shall examine the TSS to verify that it describes (for each supported platform) how the key storage functionality is invoked for each persistent secret and private key described in the TSS (it should be noted that this may be through a mechanism that is not implemented by the MDM Server; nonetheless, that mechanism will be identified in the TSS as part of this evaluation activity).

Persistent secrets and private keys manipulated by the TSF:

The evaluator reviews the TSS to determine that it makes a case that, for each item listed as being manipulated by the TOE, it is not written unencrypted to persistent memory, and that the item is stored by the platform.

Guidance
TBD

Tests
TBD

FCS_STG_EXT.2 Encrypted Cryptographic Key Storage

This is a selection-based component. Its inclusion depends upon selection from FCS_STG_EXT.1.1.

FCS_STG_EXT.2.1

The TSF shall [**selection**: *invoke platform-provided functionality, implement functionality*] to encrypt all keys using AES in the [**selection**: *Key Wrap (KW) mode, Key Wrap with Padding (KWP) mode, GCM, CCM, CBC mode*].

Application Note: This requirement states that keys used by the TSF shall not be kept in plaintext. The intent of this requirement is to ensure that the private keys, credentials, and persistent secrets cannot be accessed in the TOE in an unencrypted state, allowing an attacker to access keys without having to exhaust the AES key space.

This requirement must be included in the ST if the selection in FCS_STG_EXT.1 indicates that the TSF is protecting private keys and persistent secrets with encryption rather than the platform-provided key storage.

Evaluation Activities

FCS_STG_EXT.2
TSS
The evaluator shall examine the TSS to ensure it describes in detail how user credentials, persistent secret and private keys are stored and encrypted. The evaluator shall review the TSS to determine that it makes a case that key material is not written unencrypted to persistent memory and that it identifies the mode of encryption.

If "invoke platform-provided functionality" is selected, the evaluator shall examine the TSS to verify that it describes (for each supported platform) how the key encryption functionality is invoked (it should be noted that this may be through a mechanism that is not implemented by the MDM server; nonetheless, that mechanism will be identified in the TSS as part of this evaluation activity).

Guidance
TBD

Tests
TBD

4.1.5 Identification and Authentication (FIA)

FIA_CLI_EXT.1 Client Authorization

FIA_CLI_EXT.1.1

The TSF shall require a unique [**selection**: *certificate, token as defined in FIA_TOK_EXT.1*] for each client device.

Application Note: Token is used here in a generic way to be some form of unique identifier that is not certificate-based as defined in FIA_TOK_EXT.1. If "token as defined in FIA_TOK_EXT.1" is selected, FIA_TOK_EXT.1 must be included in the ST.

Evaluation Activities

FIA_CLI_EXT.1
TSS
The evaluator shall ensure that the TSS describes how the client is uniquely identified.

Guidance
None.

Tests
None.

FIA_ENR_EXT.1 Enrollment of Mobile Device into Management

FIA_ENR_EXT.1.1

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

Application Note: The MDM server may use its own directory or a directory server to perform the authentication decision for users performing the remote enrollment of a MD.

FIA_ENR_EXT.1.2

The TSF shall limit the user's enrollment of devices to devices specified by [**selection**: *IMEI*, *[**assignment**: a unique device ID]*] and [**selection**: *specific device models*, *a number of devices*, *specific time period*, *[**assignment**: other features]*, *no other features*].

Application Note: This requirement is designed to permit the enterprise to restrict users' enrollment of devices. A unique device ID is required to limit the user's enrollment. The unique device ID can be the IMEI or an ID specific to a particular platform.

Evaluation Activities

FIA_ENR_EXT.1.1
TSS
The evaluator shall examine the TSS and verify that it describes the process of enrollment for each MDM agent or platform listed as supported in the ST. This description shall include the trusted path used for enrollment (FTP_TRP.1/TRUSTPATH_ENROLL), the method of user authentication (username or password, token, etc.), the method of authentication decision (local or remote authentication services), and the actions performed on the MDM server upon successful authentication.

Guidance
TBD

Tests
- *Test FIA_ENR_EXT.1.1:1: The evaluator shall attempt to enroll a device without providing correct credentials. The evaluator shall verify that the device is not enrolled and that the described enrollment actions are not taken.*
- *Test FIA_ENR_EXT.1.1:2: The evaluator shall attempt to enroll the device providing correct credentials. The evaluator shall verify that the device is enrolled and that the described enrollment actions are taken.*

FIA_ENR_EXT.1.2
TSS
The evaluator shall examine the TSS and verify that it implements a policy to limit the user's enrollment of devices.

Guidance
The evaluator shall ensure that the administrative guidance describes the methods of restricting user enrollment and that it instructs the administrator how to configure the restrictions.

Tests
For each type of policy selected, the evaluator shall perform the following:
- *Test FIA_ENR_EXT.1.2:1: The evaluator shall attempt to configure the MDM server according to the administrative guidance in order to prevent enrollment. The evaluator shall verify that the user cannot enroll a device outside of the configured limitation. (For example, the evaluator may try to enroll a disallowed device, or may try to enroll additional devices beyond the number allowed.)*

FIA_TOK_EXT.1 Client Tokens

This is a selection-based component. Its inclusion depends upon selection from

***FIA_CLI_EXT.1.1*.**

FIA_TOK_EXT.1.1

The TSF shall [**selection**: *invoke platform-provided functionality, implement functionality*] to use [**selection**: *IMEI, [**assignment**: other unique device ID]*] to generate a unique token for each client device.

Application Note: This SFR is included in the ST if "token as defined in FIA_TOK_EXT.1" is selected in FIA_CLI_EXT.1.

Evaluation Activities

FIA_TOK_EXT.1
TSS
The evaluator shall review the TSS and verify that the TSF uses either unique identifiers from the client device or a server-specific mechanism to generate a unique token that will be used for verifying the identity of the client device. If the server generates the token using cryptographic functions, it must use algorithms in FCS_COP.1(ANY) (specific algorithms as needed by the vendor).

If "invoke platform-provided functionality" is selected, the evaluator shall examine the TSS to verify that it describes (for each supported platform) how this functionality is invoked (it should be noted that this may be through a mechanism that is not implemented by the MDM server; nonetheless, that mechanism will be identified in the TSS as part of this evaluation activity).

If "implement functionality" is selected, the evaluator shall examine the TSS to verify that it describes the methods to generate the token.

Guidance
None.

Tests
For each MDM agent or platform listed as supported in the ST:

- *Test FIA_TOK_EXT.1:1: The evaluator shall use appropriate combinations of specialized operational environment and development tools (debuggers, simulators, etc.) for the TOE and instrumented TOE builds as needed to perform this test.*
- *Test FIA_TOK_EXT.1:2: The evaluator shall concurrently enroll 10 devices and ensure that the token for each is unique, per the methods described in the TSS.*

FIA_UAU.1 Timing of Authentication

FIA_UAU.1.1

The TSF shall [**selection**: ***invoke platform-provided functionality*** , ***implement functionality***] **to** allow [**assignment**: *list of TSF mediated actions*] on behalf of the user to be performed before the user is authenticated **with the server**.

FIA_UAU.1.2

The TSF shall [**selection**: ***invoke platform-provided functionality***, ***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.

Application Note: This requirement ensures that any user attempting to access the TSF must be authenticated. These users may be administrators attempting to administer the TOE or ordinary users attempting to enroll for management by the MDM system. The ST author is responsible for assigning the list of actions that can take place before this authentication. The TSF or TOE platform may use enterprise authentication to meet this requirement.

For distributed TOEs, at least one TOE component has to support the authentication of administrators but not necessarily all TOE components. In case not all TOE components support authentication for administrators the TSS must describe how administrators are authenticated and identified.

Evaluation Activities

FIA_UAU.1
TSS

If "invoke platform-provided functionality" is selected, the evaluator shall examine the TSS to verify that it describes (for each supported platform) how this functionality is invoked (it should be noted that this may be through a mechanism that is not implemented by the MDM server; nonetheless, that mechanism will be identified in the TSS as part of this evaluation activity).

Guidance
TBD

Tests
The evaluator shall perform the following tests:

- *Test FIA_UAU.1:1: The evaluator shall attempt to perform the prohibited actions before authentication. The evaluator shall verify the actions cannot be performed.*
- *Test FIA_UAU.1:2: The evaluator shall attempt to perform the prohibited actions after authentication. The evaluator shall verify the actions can be performed.*

FIA_UAU.4 Single-Use Authentication Mechanisms

This is an objective component.

FIA_UAU.4.1

The TSF shall prevent reuse of authentication data related to [**assignment**: *identified authentication mechanisms*].

Application Note: This requirement references the authentication mechanisms used to authenticate the user for enrollment in FIA_ENR_EXT.1.1. If a username and password is used to authenticate the user for enrollment, the password must not be reused. Thus if the user has two devices enrolled in management or needs to re-enroll the same device (i.e., after a device wipe), the password must be different for each enrollment. Additionally, if two different users are enrolling the password must be different for each user.

Evaluation Activities

FIA_UAU.4
TSS
The evaluator shall verify that the TSS contains a description of the process of enrollment for each MDM agent or platform listed as supported in the ST. This description shall include the method of user authentication (username or password, token, etc.) and how reuse of the authentication data is prevented.

Guidance
The evaluator shall ensure that the administrative guidance describes the methods of restricting user enrollment and that it instructs the administrator on how to configure the restrictions.

Tests
- *Test FIA_UAU.4:1: The evaluator shall enroll a device providing correct credentials. The evaluator shall attempt to enroll a second device using the same credentials used to enroll the first device. The evaluator shall verify that the second device could not enroll.*

FIA_X509_EXT.1/CERTVAL_MAN X.509 Certificate Validation

FIA_X509_EXT.1.1/CERTVAL_MAN

The TSF shall TD0641 was deliberately not applied because we anticipate the X509 Package will be referenced instead. [**selection**: *invoke platform-provided functionality*, *implement functionality*] to validate certificates in accordance with the following rules:

- RFC 5280 certificate validation and certificate path validation.
- The certificate path must terminate with a trusted CA certificate.
- The TSF shall validate a certificate path by ensuring the presence of the basicConstraints extension, 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 [**selection**: *OCSP as specified in RFC 6960*, *CRL as specified in RFC 5280 Section 6.3*, *a CRL as specified in RFC 5759 Section 5*, *an OCSP TLS Status*

Request Extension (i.e., OCSP stapling) as specified in RFC 6066, OCSP TLS Multi-Certificate Status Request Extension (i.e., OCSP Multi-Stapling) as specified in RFC 6961].

- 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: FIA_X509_EXT.1.1/CERTVAL_MAN lists the rules for validating certificates. The ST author selects whether revocation status is verified using OCSP or CRLs. FIA_X509_EXT.2 requires that certificates are used for trusted channels; this use requires that the extendedKeyUsage rules are verified. Certificates may optionally be used for code signing and policy signing and, if implemented, must be validated to contain the corresponding extendedKeyUsage.

OCSP stapling and OCSP multi-stapling only support TLS server certificate validation. If other certificate types are validated, either OCSP or CRL should be claimed. If OCSP is not supported the EKU provision for checking the OCSP Signing purpose is met by default.

Regardless of the selection of *implement functionality* or *invoke platform-provided functionality*, the validation is expected to end in a trusted root CA certificate in a root store managed by the platform.

FIA_X509_EXT.1.2/CERTVAL_MAN

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

Application Note: This requirement applies to certificates that are used and processed by the TOE or platform and restricts the certificates that may be added as trusted CA certificates.

Evaluation Activities

FIA_X509_EXT.1.1/CERTVAL_MAN
TSS
If "invoke platform-provided functionality" is selected:

The evaluator shall examine the TSS to verify that it describes (for each supported platform) how this functionality is invoked (it should be noted that this may be through a mechanism that is not implemented by the MDM server; nonetheless, that mechanism will be identified in the TSS as part of this evaluation activity).

The TSS must describe when revocation checking is performed. It is expected that revocation checking is performed when a certificate is used in an authentication step. It is not sufficient to verify the status of an X.509 certificate only when it is loaded onto the device.

If "implement functionality" is selected:

The evaluator shall ensure the TSS describes where the check of validity of the certificates takes place. The evaluator ensures the TSS also provides a description of the certificate path validation algorithm.

The TSS must describe when revocation checking is performed. It is expected that revocation checking is performed when a certificate is used in an authentication step. It is not sufficient to verify the status of an X.509 certificate only when it is loaded onto the device.

Guidance
TBD

Tests
If "implement functionality" is selected:

The tests described must be performed in conjunction with the other certificate services evaluation activities, including each of the functions in FIA_X509_EXT.2.1. The tests for the extendedKeyUsage rules are performed in conjunction with the uses that require those rules. The evaluator shall create a chain of at least three certificates: the node certificate to be tested, an Intermediate CA, and the self-signed Root CA.

- *Test FIA_X509_EXT.1.1/CERTVAL_MAN:1: The evaluator shall demonstrate that validating a certificate without a valid certification path results in the function failing, for each of the following reasons, in turn:*
  - *by establishing a certificate path in which one of the issuing certificates is not a CA certificate,*
  - *by omitting the basicConstraints field in one of the issuing certificates,*
  - *by setting the basicConstraints field in an issuing certificate to have CA=False,*
  - *by omitting the CA signing bit of the key usage field in an issuing certificate, and*
  - *by setting the path length field of a valid CA field to a value strictly less than the certificate path.*

  The evaluator shall then establish a valid certificate path consisting of valid CA certificates, and demonstrate that the function succeeds. The evaluator shall then remove trust in one of the CA certificates, and show that the function fails.
- *Test FIA_X509_EXT.1.1/CERTVAL_MAN:2: The evaluator shall demonstrate that validating an expired certificate results in the function failing.*
- *Test FIA_X509_EXT.1.1/CERTVAL_MAN:3: The evaluator shall test that the TOE can properly handle revoked certificates--conditional on whether CRL, OCSP or OCSP stapling is selected; if multiple methods are selected, then a test shall be performed for each method. The evaluator shall test revocation of the node certificate and revocation of the intermediate CA certificate (i.e., the intermediate CA certificate should be revoked by the root CA). If OCSP stapling per RFC 6066 is the only supported revocation method, testing revocation of the intermediate CA certificate is omitted. The evaluator shall ensure that a valid certificate is used, and that the validation function succeeds. The evaluator then attempts the test with a certificate that has been revoked (for each method chosen in the selection) to ensure when the certificate is no longer valid that the validation function fails.*
- *Test FIA_X509_EXT.1.1/CERTVAL_MAN:4: If OCSP option is selected, the evaluator shall send the TOE an OCSP response signed by a certificate that does not have the OCSP signing purpose and verify that validation of the OCSP response fails. If CRL is selected, the evaluator shall cause a CA to sign a CRL with a certificate that has a key usage extension but does not have the cRLsign key usage bit set, and verify that validation of the CRL fails.*
- *Test FIA_X509_EXT.1.1/CERTVAL_MAN:5: The evaluator shall modify any byte in the first eight bytes of the certificate and demonstrate that the certificate fails to validate. (The certificate will fail to parse correctly.)*
- *Test FIA_X509_EXT.1.1/CERTVAL_MAN:6: The evaluator shall modify any byte in the last byte of the certificate and demonstrate that the certificate fails to validate. (The signature on the certificate will not validate.)*
- *Test FIA_X509_EXT.1.1/CERTVAL_MAN:7: The evaluator shall modify any byte in the public key of the certificate and demonstrate that the certificate fails to validate. (The signature on the certificate will not validate.)*
- *Test FIA_X509_EXT.1.1/CERTVAL_MAN:8:*
  - *Test FIA_X509_EXT.1.1/CERTVAL_MAN:8.1: Test 8a: (Conditional on support for EC certificates as indicated in FCS_COP.1/SIGN_ALG). The evaluator shall establish a valid, trusted certificate chain consisting of an EC leaf certificate, an EC Intermediate CA certificate not designated as a trust anchor, and an EC certificate designated as a trusted anchor, where the elliptic curve parameters are specified as a named curve. The evaluator shall confirm that the TOE validates the certificate chain.*
  - *Test FIA_X509_EXT.1.1/CERTVAL_MAN:8.2: Test 8b: (Conditional on support for EC certificates as indicated in FCS_COP.1/SIGN_ALG). The evaluator shall replace the intermediate certificate in the certificate chain for Test 8a with a modified certificate, where the modified intermediate CA has a public key information field where the EC parameters uses an explicit format version of the Elliptic Curve parameters in the public key information field of the intermediate CA certificate from Test 8a, and the modified Intermediate CA certificate is signed by the trusted EC root CA, but having no other changes. The evaluator shall confirm the TOE treats the certificate as invalid.*

FIA_X509_EXT.1.2/CERTVAL_MAN
TSS
If "invoke platform-provided functionality" is selected:

The evaluator shall examine the TSS to verify that it describes (for each supported platform) how this functionality is invoked (it should be noted that this may be through a mechanism that is not implemented by the MDM server; nonetheless, that mechanism will be identified in the TSS as part of this evaluation activity).

Guidance
TBD

Tests
If "implement functionality" is selected:

The tests described must be performed in conjunction with the other certificate services evaluation activities, including the functions in FIA_X509_EXT.2.1. The evaluator shall create a chain of at least four certificates: the node certificate to be tested, two Intermediate CAs, and the self-signed Root CA.

- *Test FIA_X509_EXT.1.2/CERTVAL_MAN:1: The evaluator shall construct a certificate path, such that the certificate of the CA issuing the TOEs certificate does not contain the basicConstraints extension. The validation of the certificate path fails.*
- *Test FIA_X509_EXT.1.2/CERTVAL_MAN:2: The evaluator shall construct a certificate path, such that the certificate of the CA issuing the TOEs certificate has the cA flag in the basicConstraints extension not set. The validation of the certificate path fails.*
- *Test FIA_X509_EXT.1.2/CERTVAL_MAN:3: The evaluator shall construct a certificate path, such that the certificate of the CA issuing the TOEs certificate has the cA flag in the basicConstraints extension set to TRUE. The validation of the certificate path succeeds.*

FIA_X509_EXT.1/CERTVAL_SEL X.509 Certificate Validation

This is a selection-based component. Its inclusion depends upon selection from FPT_ITT.1.1/INTER_XFER.

FIA_X509_EXT.1.1/CERTVAL_SEL

The TSF shall TD0641 was deliberately not applied because we anticipate the X509 Package will be referenced instead. [**selection**: *invoke platform-provided functionality*, *implement functionality*] to validate certificates in accordance with the following rules:

- RFC 5280 certificate validation and certificate path validation.
- The certificate path must terminate with a trusted CA certificate.
- The TSF shall validate a certificate path by ensuring the presence of the basicConstraints extension, 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 [**selection**: *OCSP as specified in RFC 6960*, *CRL as specified in RFC 5280 Section 6.3*, *a CRL as specified in RFC 5759 Section 5*, *an OCSP TLS Status Request Extension (i.e., OCSP stapling) as specified in RFC 6066*, *OCSP TLS Multi-Certificate Status Request Extension (i.e., OCSP Multi-Stapling) as specified in RFC 6961*, *no revocation method*].
- 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: FIA_X509_EXT.1.1/CERTVAL_SEL should be chosen if the TOE is distributed and the protocols selected in FPT_ITT.1/INTER_XFER use X.509 certificates for peer authentication. In this case, the use of revocation list checking is optional as there are additional requirements surrounding the enabling and disabling of the ITT channel as defined in FCO_CPC_EXT.1. If revocation checking is not supported, the ST author should select "no revocation method." However, if certificate revocation checking is supported, the ST author selects whether this is performed using OCSP or CRLs.

This SFR lists the rules for validating certificates. The ST author selects whether revocation status is verified using OCSP or CRLs. FIA_X509_EXT.2 requires that certificates are used for trusted channels; this use requires that the extendedKeyUsage rules are verified. Certificates may optionally be used for

code signing and policy signing and, if implemented, must be validated to contain the corresponding extendedKeyUsage.

OCSP stapling and OCSP multi-stapling only support TLS server certificate validation. If other certificate types are validated, either OCSP or CRL should be claimed. If OCSP is not supported the EKU provision for checking the OCSP Signing purpose is met by default.

Regardless of the selection of *implement functionality* or *invoke platform-provided functionality* , the validation is expected to end in a trusted root CA certificate in a root store managed by the platform.

FIA_X509_EXT.1.2/CERTVAL_SEL

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

Application Note: This requirement applies to certificates that are used and processed by the TOE or platform and restricts the certificates that may be added as trusted CA certificates.

Evaluation Activities

FIA_X509_EXT.1.1/CERTVAL_SEL
TSS
If "invoke platform-provided functionality" is selected:

The evaluator shall examine the TSS to verify that it describes (for each supported platform) how this functionality is invoked (it should be noted that this may be through a mechanism that is not implemented by the MDM server; nonetheless, that mechanism will be identified in the TSS as part of this evaluation activity).

The TSS must describe when revocation checking is performed. It is expected that revocation checking is performed when a certificate is used in an authentication step. It is not sufficient to verify the status of an X.509 certificate only when it is loaded onto the device. If "implement functionality" is selected:

The evaluator shall ensure the TSS describes where the check of validity of the certificates takes place. The evaluator ensures the TSS also provides a description of the certificate path validation algorithm.

The TSS must describe when revocation checking is performed. It is expected that revocation checking is performed when a certificate is used in an authentication step. It is not sufficient to verify the status of an X.509 certificate only when it is loaded onto the device.

Guidance
TBD

Tests
If "implement functionality" is selected:

The tests described must be performed in conjunction with the other certificate services evaluation activities, including each of the functions in FIA_X509_EXT.2.1. The tests for the extendedKeyUsage rules are performed in conjunction with the uses that require those rules. The evaluator shall create a chain of at least three certificates: the node certificate to be tested, an Intermediate CA, and the self-signed Root CA.

- *Test FIA_X509_EXT.1.1/CERTVAL_SEL:1: The evaluator shall demonstrate that validating a certificate without a valid certification path results in the function failing, for each of the following reasons, in turn:*
  - *by establishing a certificate path in which one of the issuing certificates is not a CA certificate,*
  - *by omitting the basicConstraints field in one of the issuing certificates,*
  - *by setting the basicConstraints field in an issuing certificate to have CA=False,*
  - *by omitting the CA signing bit of the key usage field in an issuing certificate, and*
  - *by setting the path length field of a valid CA field to a value strictly less than the certificate path.*

  The evaluator shall then establish a valid certificate path consisting of valid CA certificates, and demonstrate that the function succeeds. The evaluator shall then remove trust in one of the CA certificates, and show that the function fails.
- *Test FIA_X509_EXT.1.1/CERTVAL_SEL:2: The evaluator shall demonstrate that validating an expired certificate results in the function failing.*
- *Test FIA_X509_EXT.1.1/CERTVAL_SEL:3: The evaluator shall test that the TOE can properly handle revoked certificates--conditional on whether CRL, OCSP or OCSP stapling is selected; if multiple methods are selected, then a test shall be performed for each*

method. The evaluator shall test revocation of the node certificate and revocation of the intermediate CA certificate (i.e., the intermediate CA certificate should be revoked by the root CA). If OCSP stapling per RFC 6066 is the only supported revocation method, testing revocation of the intermediate CA certificate is omitted. The evaluator shall ensure that a valid certificate is used, and that the validation function succeeds. The evaluator then attempts the test with a certificate that has been revoked (for each method chosen in the selection) to ensure when the certificate is no longer valid that the validation function fails.

- *Test FIA_X509_EXT.1.1/CERTVAL_SEL:4: If OCSP option is selected, the evaluator shall send the TOE an OCSP response signed by a certificate that does not have the OCSP signing purpose and verify that validation of the OCSP response fails. If CRL is selected, the evaluator shall cause a CA to sign a CRL with a certificate that has a key usage extension but does not have the cRLsign key usage bit set, and verify that validation of the CRL fails.*
- *Test FIA_X509_EXT.1.1/CERTVAL_SEL:5: The evaluator shall modify any byte in the first eight bytes of the certificate and demonstrate that the certificate fails to validate. (The certificate will fail to parse correctly.)*
- *Test FIA_X509_EXT.1.1/CERTVAL_SEL:6: The evaluator shall modify any byte in the last byte of the certificate and demonstrate that the certificate fails to validate. (The signature on the certificate will not validate.)*
- *Test FIA_X509_EXT.1.1/CERTVAL_SEL:7: The evaluator shall modify any byte in the public key of the certificate and demonstrate that the certificate fails to validate. (The signature on the certificate will not validate.)*
- *Test FIA_X509_EXT.1.1/CERTVAL_SEL:8:*

  Test 8a: (Conditional on support for EC certificates as indicated in FCS_COP.1/SIGN_ALG). The evaluator shall establish a valid, trusted certificate chain consisting of an EC leaf certificate, an EC Intermediate CA certificate not designated as a trust anchor, and an EC certificate designated as a trusted anchor, where the elliptic curve parameters are specified as a named curve. The evaluator shall confirm that the TOE validates the certificate chain.

  Test 8b: (Conditional on support for EC certificates as indicated in FCS_COP.1/SIGN_ALG). The evaluator shall replace the intermediate certificate in the certificate chain for Test 8a with a modified certificate, where the modified intermediate CA has a public key information field where the EC parameters uses an explicit format version of the Elliptic Curve parameters in the public key information field of the intermediate CA certificate from Test 8a, and the modified Intermediate CA certificate is signed by the trusted EC root CA, but having no other changes. The evaluator shall confirm the TOE treats the certificate as invalid.

FIA_X509_EXT.1.2/CERTVAL_SEL
TSS
If "invoke platform-provided functionality" is selected:

The evaluator shall examine the TSS to verify that it describes (for each supported platform) how this functionality is invoked (it should be noted that this may be through a mechanism that is not implemented by the MDM server; nonetheless, that mechanism will be identified in the TSS as part of this evaluation activity).

Guidance
TBD

Tests
If "implement functionality" is selected:

The tests described must be performed in conjunction with the other certificate services evaluation activities, including the functions in FIA_X509_EXT.2.1. The evaluator shall create a chain of at least four certificates: the node certificate to be tested, two Intermediate CAs, and the self-signed Root CA.

- *Test FIA_X509_EXT.1.2/CERTVAL_SEL:1: The evaluator shall construct a certificate path, such that the certificate of the CA issuing the TOEs certificate does not contain the basicConstraints extension. The validation of the certificate path fails.*
- *Test FIA_X509_EXT.1.2/CERTVAL_SEL:2: The evaluator shall construct a certificate path, such that the certificate of the CA issuing the TOEs certificate has the cA flag in the basicConstraints extension not set. The validation of the certificate path fails.*
- *Test FIA_X509_EXT.1.2/CERTVAL_SEL:3: The evaluator shall construct a certificate path, such that the certificate of the CA issuing the TOEs certificate has the cA flag in the basicConstraints extension set to TRUE. The validation of the certificate path succeeds.*

FIA_X509_EXT.2 X.509 Certificate Authentication

FIA_X509_EXT.2.1

The TSF shall [**selection**:

- *invoke platform-provided functionality to use X.509v3 certificates as defined by RFC 5280 to support authentication for [**selection**: IPsec,*

*HTTPS, TLS, DTLS, SSH, no protocols], and [**selection**:*
- *code signing for system software updates*
- *code signing for integrity verification*
- *policy signing*
- *[**assignment**: other uses]*
- *no additional uses*

]

- *implement functionality to use X.509v3 certificates as defined by RFC 5280 to support authentication for [**selection**:*
  - *IPsec as defined in the PP-Module for VPN Client*
  - *HTTPS in accordance with FCS_HTTPS_EXT.1*
  - *TLS as defined in the Package for Transport Layer Security*
  - *DTLS as defined in the Package for Transport Layer Security*
  - *SSH as defined in the Functional Package for Secure Shell*
  - *no protocols*

  *], and [**selection**:*
  - *code signing for system software updates*
  - *code signing for integrity verification*
  - *policy signing*
  - *[**assignment**: other uses]*
  - *no additional uses*

  ].

]

Application Note: The ST author's selections must match the selection of FTP_TRP.1/TRUSTPATH_ENROLL, FTP_ITC.1/INTER_XFER_IT, FTP_ITC.1/INTER_TSF_XFER_AGENT, FPT_ITT.1/INTER_XFER, and FPT_ITT.1/INTER_XFER_AGENT. Certificates may optionally be used for trusted updates of system software (FPT_TUD_EXT.1.3) and software integrity verification (FPT_TST_EXT.1.2). If some authentication services are provided by the TOE and others by the platform, the ST author must clearly identify which services are provided by the TOE and which by the platform.

If code signing for integrity verification is selected, the MDM vendor is not expected to digitally sign DLLs from other vendors that have been incorporated into their product.

FIA_X509_EXT.2.2

When the TD0641 was deliberately not applied because we anticipate the X509 Package will be referenced instead. [**selection**: *TSF*, *TOE platform*] cannot establish a connection to determine the validity of a certificate, the TSF shall [**selection**: *invoke platform-provided functionality*, *implement functionality*] to [**selection**: *allow the administrator to choose whether to accept the certificate in these cases*, *accept the certificate*, *not accept the certificate*].

Application Note: If a valid OCSP response is not provided by the server during a TLS handshake, to include when the TOE does not support OCSP stapling, a connection may need to be established to perform a verification of the revocation status of a certificate - either to download a current, valid CRL or to perform OCSP. The selection is used to describe the behavior in the event that such a connection cannot be established (for example, due to a network error). If the TOE has determined the certificate is valid according to all other rules in FIA_X509_EXT.1/CERTVAL_MAN, the behavior indicated in the second selection must determine the validity. The TOE must not accept the certificate if it fails any of the other validation rules in FIA_X509_EXT.1/CERTVAL_MAN or if valid revocation information (a cached CRL, or recent or stapled OCSP response) indicates the certificate has been revoked. If the administrator-configured option is selected by the ST Author, the ST Author must also select function d in FMT_SMF.1/SERVER_CONF_SERVER.

If the TOE is distributed and FIA_X509_EXT.1/CERTVAL_SEL is selected, then certificate revocation checking is optional. This is due to additional authorization actions being performed in the enabling and disabling of the intra-TOE trusted channel as defined in FCO_CPC_EXT.1. In this case, a connection is not required to determine certificate validity and this SFR is trivially satisfied.

Evaluation Activities

FIA_X509_EXT.2
TSS

If "invoke platform-provided functionality" is selected, the evaluator shall examine the TSS to verify that it describes (for each supported platform) how this functionality is invoked (it should be noted that this may be through a mechanism that is not implemented by the MDM server; nonetheless, that mechanism will be identified in the TSS as part of this evaluation activity).

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

If "invoke platform-provided functionality" is selected, the evaluator shall examine the TSS to verify that it describes (for each supported platform) how this functionality is invoked (it should be noted that this may be through a mechanism that is not implemented by the MDM server; nonetheless, that mechanism will be identified in the TSS as part of this evaluation activity).

If "implement functionality" is selected, the evaluator shall examine the TSS to confirm that it describes the behavior of the TOE when a connection cannot be established during the validity check of a certificate used in establishing a trusted channel. The evaluator shall verify that any distinctions between trusted channels are described.

Guidance
There are no additional Guidance evaluation activities for this component.
If the requirement that the administrator is able to specify the default action is selected, then the evaluator shall ensure that the operational guidance contains instructions on how this configuration action is performed.

Tests
There are no test activities for this component.
The evaluator shall perform the following test for each trusted channel:

- *Test FIA_X509_EXT.2:1: The evaluator shall demonstrate use of a valid certificate requiring certificate validation checking to be performed in at least some part by communicating with a non-TOE IT entity. The evaluator shall then manipulate the environment so that the TOE is unable to verify the validity of the certificate, and observe that the action selected in FIA_X509_EXT.2.2 is performed. If the selected action is administrator-configurable, then the evaluator shall follow the operational guidance to determine that all supported administrator-configurable options behave in their documented manner.*

FIA_X509_EXT.3 X.509 Enrollment

This is an objective component.

FIA_X509_EXT.3.1

covered by X509_EXT.3 in the X.509 package, this will be removed The TSF shall [**selection**: *invoke platform-provided functionality*, *implement functionality*] to generate a Certificate Request Message as specified by RFC 2986 and be able to provide the following information in the request: public key and [**selection**: *device-specific information*, *Common Name*, *Organization*, *Organizational Unit*, *Country*].

Application Note: The public key is the public key portion of the public-private key pair generated by the TOE as specified in FCS_CKM.1.1.

As Enrollment over Secure Transport (EST) is a new standard that has not yet been widely adopted, this requirement is included as an interim objective requirement in order to allow developers to distinguish those products which have to have the ability to generate Certificate Request Messages but do not yet implement EST.

FIA_X509_EXT.3.2

The TSF shall [**selection**: *invoke platform-provided functionality*, *implement functionality*] to validate the chain of certificates from the Root CA upon receiving the CA Certificate Response.

Evaluation Activities

FIA_X509_EXT.3
TSS
If "invoke platform-provided functionality" is selected, the evaluator shall examine the TSS to verify that it describes (for each supported platform) how this functionality is invoked (it should be noted that this may be through a mechanism that is not implemented by the MDM server; nonetheless, that mechanism will be identified in the TSS as part of this evaluation activity).

If the ST author selects "device-specific information," the evaluator shall verify that the TSS contains a description of the device-specific fields used in certificate requests.

If "invoke platform-provided functionality" is selected, the evaluator shall examine the TSS to verify that it describes (for each supported platform) how this functionality is invoked (it should be noted that this may be through a mechanism that is not implemented by the MDM server; nonetheless, that mechanism will be identified in the TSS as part of this evaluation activity).

Guidance
There are no additional Guidance evaluation activities for this component.
The evaluator shall check to ensure that the operational guidance contains instructions on requesting certificates from a CA, including generation of a Certificate Request Message. If the ST author selects "Common Name," "Organization," "Organizational Unit," or "Country," the evaluator shall ensure that this guidance includes instructions for establishing these fields before creating the certificate request message.

Tests
There are no test activities for this component.

- *Test FIA_X509_EXT.3:1: The evaluator shall use the operational guidance to cause the TOE to generate a certificate request message. The evaluator shall capture the generated message and ensure that it conforms to the format specified. The evaluator shall confirm that the certificate request provides the public key and other required information, including any necessary user-input information.*
- *Test FIA_X509_EXT.3:2: The evaluator shall demonstrate that validating a certificate response message without a valid certification path results in the function failing. The evaluator shall then load a certificate or certificates as trusted CAs needed to validate the certificate response message, and demonstrate that the function succeeds. The evaluator shall then delete one of the certificates, and show that the function fails.*

FIA_X509_EXT.4 Alternate X.509 Enrollment

This is an objective component.

FIA_X509_EXT.4.1

covered by ESTC_EXT.1 in the X.509 package, this will be removed The TSF shall use the Enrollment over Secure Transport (EST) protocol as specified in RFC 7030 to request certificate enrollment using the simple enrollment method described in RFC 7030 Section 4.2.

FIA_X509_EXT.4.2

The TSF shall be capable of authenticating EST requests using an existing certificate and corresponding private key as specified by RFC 7030 Section 3.3.2.

FIA_X509_EXT.4.3

The TSF shall be capable of authenticating EST requests using HTTP Basic Authentication with a username and password as specified by RFC 7030 Section 3.2.3.

FIA_X509_EXT.4.4

The TSF shall perform authentication of the EST server using an Explicit Trust Anchor following the rules described in RFC 7030, section 3.6.1.

Application Note: EST also uses HTTPS as specified in FCS_HTTPS_EXT.1 to establish a secure connection to an EST server, and thus, the ST author must also include FCS_HTTPS_EXT.1 in the main body of the ST. The separate Trust Anchor Database dedicated to EST operations is described as Explicit Trust Anchors in RFC 7030.

FIA_X509_EXT.4.5

The TSF shall be capable of requesting server-provided private keys as specified in RFC 7030 Section 4.4.

FIA_X509_EXT.4.6

The TSF shall be capable of updating its EST-specific Trust Anchor Database using the "Root CA Key Update" process described in RFC 7030 Section 4.1.3.

FIA_X509_EXT.4.7

The TSF shall generate a Certificate Request Message for EST as specified in RFC 2986 and be able to provide the following information in the request: public key and [**selection**:

- *device-specific information*

- *Common Name, Organization, Organizational Unit, and Country*

].

FIA_X509_EXT.4.8

The TSF shall validate the chain of certificates from the Root CA certificate in the Trust Anchor Database to the EST Server CA certificate upon receiving a CA Certificates Response.

Application Note: The public key referenced in FIA_X509_EXT.4.7 is the public key portion of the public-private key pair generated by the TOE as specified in FCS_CKM.1.

Evaluation Activities

FIA_X509_EXT.4
TSS
TBD

Guidance
The evaluator shall check to ensure that the operational guidance contains instructions on requesting certificates from an EST server, including generating a Certificate Request Message.

Tests
The evaluator shall also perform the following tests. Other tests are performed in conjunction with the TLS evaluation activities.

- *Test FIA_X509_EXT.4:1: The evaluator shall use the operational guidance to cause the TOE to request certificate enrollment from an EST server using the simple enrollment method described in RFC 7030 Section 4.2, authenticating the certificate request to the server using an existing certificate and private key as described by RFC 7030 Section 3.3.2. The evaluator shall confirm that the resulting certificate is successfully obtained and installed in the TOE key store.*
- *Test FIA_X509_EXT.4:2: The evaluator shall use the operational guidance to cause the TOE to request certificate enrollment from an EST server using the simple enrollment method described in RFC 7030 Section 4.2, authenticating the certificate request to the server using a username and password as described by RFC 7030 Section 3.2.3. The evaluator shall confirm that the resulting certificate is successfully obtained and installed in the TOE key store.*
- *Test FIA_X509_EXT.4:3: The evaluator shall modify the EST server to return a certificate containing a different public key than the key included in the TOEs certificate request. The evaluator shall use the operational guidance to cause the TOE to request certificate enrollment from an EST server. The evaluator shall confirm that the TOE does not accept the resulting certificate since the public key in the issued certificate does not match the public key in the certificate request.*
- *Test FIA_X509_EXT.4:4: The evaluator shall configure the EST server or use a man-in-the-middle tool to present a server certificate to the TOE that is present in the TOE general Trust Anchor Database but not its EST-specific Trust Anchor Database. The evaluator shall cause the TOE to request certificate enrollment from the EST server. The evaluator shall verify that the request is not successful.*
- *Test FIA_X509_EXT.4:5: The evaluator shall configure the EST server or use a man-in-the-middle tool to present an invalid certificate. The evaluator shall cause the TOE to request certificate enrollment from the EST server. The evaluator shall verify that the request is not successful. The evaluator shall configure the EST server or use a man-in-the-middle tool to present a certificate that does not have the CMC RA purpose and verify that requests to the EST server fail. The tester shall repeat the test using a valid certificate and a certificate that contains the CMC RA purpose and verify that the certificate enrollment requests succeed.*
- *Test FIA_X509_EXT.4:6: The evaluator shall use a packet sniffing tool between the TOE and an EST server. The evaluator shall turn on the sniffing tool and cause the TOE to request certificate enrollment from an EST server. The evaluator shall verify that the EST protocol interaction occurs over a Transport Layer Security (TLS) protected connection. The evaluator is not expected to decrypt the connection but rather observe that the packets conform to the TLS protocol format.*
- *Test FIA_X509_EXT.4:7: The evaluator shall use the operational guidance to cause the TOE to request a server-provided private key and certificate from an EST server. The evaluator shall confirm that the resulting private key and certificate are successfully obtained and installed in the TOE key store.*
- *Test FIA_X509_EXT.4:8: The evaluator shall modify the EST server to, in response to a server-provided private key and certificate request, return a private key that does not correspond with the public key in the returned certificate. The evaluator shall use the operational guidance to cause the TOE to request a server-provided private key and certificate. The evaluator shall confirm that the TOE does not accept the resulting private key and certificate since the private key and public key do not correspond.*
- *Test FIA_X509_EXT.4:9: The evaluator shall configure the EST server to provide a "Root CA*

Key Update" as described in RFC 7030 Section 4.1.3. The evaluator shall cause the TOE to request CA certificates from the EST server and shall confirm that the EST-specific Trust Anchor Database is updated with the new trust anchor.

- *Test FIA_X509_EXT.4:10: The evaluator shall configure the EST server to provide a "Root CA Key Update" as described in RFC 7030 Section 4.1.3, but shall modify part of the NewWithOld certificate's generated signature. The evaluator shall cause the TOE to request CA certificates from the EST server and shall confirm that the EST-specific Trust Anchor Database is not updated with the new trust anchor since the signature did not verify.*
- *Test FIA_X509_EXT.4:11: The evaluator shall use the operational guidance to cause the TOE to generate a certificate request message. The evaluator shall capture the generated message and ensure that it conforms with the format specified by RFC 2986. The evaluator shall confirm that the certificate request provides the public key and other required information, including any necessary user-input information.*

4.1.6 Security Management (FMT)

FMT_MOF.1/FUNCBE Management of Security Functions Behavior

FMT_MOF.1.1/FUNCBE

The TSF shall restrict the ability to **perform** the functions

- listed in FMT_SMF.1/SERVER_CONF_AGENT
- enable, disable, and modify policies listed in FMT_SMF.1/SERVER_CONF_AGENT
- listed in FMT_SMF.1/SERVER_CONF_SERVER
- [**selection**: *enable, disable, and modify policies listed in FMT_SMF.1/MAS, no other functions*]

to [*authorized administrators*].

Application Note: This requirement outlines the functions that administrators will have the power to enable, disable, modify, and monitor functions and policies listed in FMT_SMF.1/SERVER_CONF_AGENT. It also includes functions necessary to maintain and configure the MDM server itself.

You must select "enable, disable, and modify policies listed in FMT_SMF.1/MAS" if the TOE includes MAS functionality and FMT_SMF.1/MAS, FAU_GEN.1/MAS_SERVER, FMT_MOF.1/MANAGEMENT_MAS, FMT_SMR.1/SECMAN_ROLES_MAS must be included in the ST.

If "enable, disable, and modify policies listed in FMT_SMF.1/MAS" is chosen, FMT_SMR.1.1/SECMAN_ROLES must be included in the ST.

Evaluation Activities

FMT_MOF.1/FUNCBE
TSS
The evaluator shall examine the TSS and user documents to ensure that they describe what security management functions are restricted to the administrator and what actions can be taken for each management function. The evaluator shall verify that the security management functions are restricted to authorized administrators and the administrator is only able to take the actions as described in the user documents.

Guidance
TBD

Tests

- *Test FMT_MOF.1/FUNCBE:1: The evaluator shall attempt to access the functions and policies in FMT_SMF.1/SERVER_CONF_AGENT as an unauthorized user and verify that the attempt fails.*
- *Test FMT_MOF.1/FUNCBE:2: [conditional] The evaluator shall attempt to access the functions and policies in FMT_SMF.1/MAS as an unauthorized user and verify that the attempt fails.*

FMT_MOF.1/MANAGEMENT_ENROLL Management of Security Functions Behavior (Enrollment)

FMT_MOF.1.1/MANAGEMENT_ENROLL

The **MDM server** shall restrict the ability to [***initiate the enrollment process***] to [*authorized administrators and MD users*].

Application Note: This requirement outlines the enrollment functions that both administrators and MD users may perform. The enrollment actions are identified in the TSS as a part of FIA_ENR_EXT.1.

The authorized administrator does not remotely initiate enrollment of the MDs that are in the possession of users but may enroll MDs when they are in the possession of the administrator, for example, before distributing the MDs to the users.

Evaluation Activities

FMT_MOF.1/MANAGEMENT_ENROLL
TSS
The evaluator shall examine the TSS and verify that it describes how unauthorized users are prevented from enrolling in the MDM services.

Guidance
TBD

Tests
The test of this function is performed in conjunction with FIA_ENR_EXT.1.

FMT_MOF.1/MANAGEMENT_MAS Management of Functions in (MAS Server Downloads)

This is a selection-based component. Its inclusion depends upon selection from FMT_MOF.1.1/FUNCBE.

FMT_MOF.1.1/MANAGEMENT_MAS

The **MAS Server** shall restrict the ability to [*enable, modify the behavior of*] the functions [*downloading applications*] to [*enrolled MDs that are compliant with MDM policies and assigned to a user in the application access group*].

Application Note: This requirement is claimed if "enable, disable, and modify policies listed in FMT_SMF.1/MAS" is selected in FMT_MOF.1.1/FUNCBE.

Evaluation Activities

FMT_MOF.1/MANAGEMENT_MAS
TSS
The evaluator shall examine the TSS to determine that all methods of initiating an application download or update push are specified.

Guidance
The evaluator shall confirm that the operational guidance contains how to initiate an application download or update push.

Tests
- *Test FMT_MOF.1/MANAGEMENT_MAS:1: The evaluator shall ensure that the MAS Server verifies that the MD is enrolled in the MDM Server and is in a compliant state. The evaluator shall verify that either none, or only those applications needed for enrollment, can be downloaded from the MAS Server prior to enrolling the MD with the MDM.*
- *Test FMT_MOF.1/MANAGEMENT_MAS:2: The evaluator shall ensure that the MAS Server verifies that the MD is enrolled in the MDM Server and is in a compliant state. The evaluator shall make policy changes to make the enrolled MD non-compliant. The evaluator shall verify that either none, or only those applications needed to bring the MD back into compliance, are able to be downloaded.*
- *Test FMT_MOF.1/MANAGEMENT_MAS:3: The evaluator shall ensure that the MAS Server verifies that the MD is enrolled in the MDM Server and is in a compliant state. The evaluator shall unenroll the MD and verify that an application can not be downloaded from the MAS Server after unenrollment.*

FMT_POL_EXT.1 Trusted Policy Update

FMT_POL_EXT.1.1

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

Application Note: The intent of this requirement is to cryptographically tie the policies to the enterprise that mandated the policy, not to protect the policies in transit (as they are already protected by FPT_ITT.1/INTER_XFER / FPT_ITT.1/INTER_XFER_AGENT or FTP_ITC.1/INTER_TSF_XFER_AGENT). This is especially critical for users who connect to multiple enterprises.

FMT_POL_EXT.1.2

The TSF shall sign policies and policy updates using a private key associated with [**selection**: *an X509 certificate*, *a public key provisioned to the agent*] trusted by the agent for policy verification.

Application Note: If "an X509 certificate" is selected, the "policy signing" option in FIA_X509_EXT.2 is also claimed.

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 [**selection**: *client authentication via an X509 certificate representing the agent*, *a token issued to the agent and associated with a policy signing key uniquely associated to the policy*].

Application Note: If "client authentication via an X509 certificate representing the agent" is claimed, the appropriate protocol in FIA_X509_EXT.2 is also claimed. If "a token issued to the agent and associated with a unique policy signing key uniquely associated to the policy" is claimed, FIA_TOK_EXT.1.1 is also claimed and the TSF maintains the association of the key pairs and the agent tokens. When multiple policies are supported, a unique policy signing key for each policy is used.

Evaluation Activities

FMT_POL_EXT.1
TSS
The evaluator shall verify that the ST describes how policies are signed, to include whether the private key used for signing is associated with an X509 certificate or public key, the method for distributing the policy verification material (a certificate or provisioned public key) to the agent, and the method for distinguishing whether a policy is appropriate for an agent. If tokens are claimed in FMT_POL_EXT.1.3, the evaluator shall verify the ST describes how tokens are established and distributed to the agent.

Guidance
If applicable, the evaluator shall verify that the operational guidance instructs administrators on configuring the enterprise certificate to be used for signing policies or signing the policies before applying them.

Tests
The evaluator shall perform a policy update in accordance with FMT_SMF.1/SERVER_CONF_AGENT. The evaluator shall examine the policy either at the MDM server, in transmission, or at the MDM agent, and verify the TSF signs the update and provides it to the MDM agent.

FMT_SAE_EXT.1 Security Attribute Expiration

This is an objective component.

FMT_SAE_EXT.1.1

The TSF shall be able to specify a configurable expiration time for enrollment authentication data.

FMT_SAE_EXT.1.2

The TSF shall be able to deny enrollment after the expiration time for the enrollment authentication data has passed.

Application Note: This requirement references the user authenticator used for device enrollment in management in FIA_ENR_EXT.1.1. The user authenticator must only be valid for a configurable time limit. If the authenticator is expired, even if entered correctly, enrollment must not occur.

The length of the time the authenticator is valid for is configured per function c.5 in FMT_SMF.1/SERVER_CONF_SERVER. If FMT_SAE_EXT.1 is included in the ST, then function g must be selected in FMT_SMF.1/SERVER_CONF_SERVER.

Evaluation Activities

FMT_SAE_EXT.1
TSS
The evaluator shall verify that the TSS contains a description of the process of enrollment for each MDM agent or platform listed as supported in the ST. This description shall be the method of user authentication (username or password, token, etc.).

Guidance
The evaluator shall check to ensure that the operational guidance contains instructions to configure the expiration time for each method of user authentication listed in the TSS.

Tests
- *Test FMT_SAE_EXT.1:1: The evaluator shall configure the MDM server according to the administrative guidance to set an expiration time for the enrollment authentication data. For each method of user authentication listed in the TSS, the evaluator shall attempt to enroll using authentication data that has expired. The evaluator shall verify that enrollment was unsuccessful.*

FMT_SMF.1/MAS Specification of Management Functions (MAS Server)

This is a selection-based component. Its inclusion depends upon selection from FMT_MOF.1.1/FUNCBE.

FMT_SMF.1.1/MAS

The **MAS Server** shall be capable of performing the following management functions: [
- *Configure application access groups*
- *Download applications*
- *[**selection**: [**assignment**: other MAS management functions] , no other functions]*

]

Application Note: This requirement captures all the configuration functionality in the MAS Server to configure the underlying MAS Server. The ST author can add more commands and configuration policies by completing the assignment statement.

The MAS Server must be able to create groups to configure which applications a user can access based on which group they are in. If the MAS Server uses the groups defined by the MDM, then it must communicate with the MDM server (if separate server) to determine which applications the user can access.

This requirement is claimed if "enable, disable, and modify policies listed in FMT_SMF.1/MAS" is selected in FMT_MOF.1.1/FUNCBE.

Evaluation Activities

FMT_SMF.1/MAS
TSS
The evaluator shall examine the TSS to ensure that it describes each management function listed.

The evaluator shall examine the TSS to determine if the MAS Server creates its own groups or relies on the groups specified by the MDM server.

Guidance
The evaluator shall confirm that the operational guidance contains how to create and define user groups and how to specify which applications are accessible by which group.

The evaluator shall verify the operational guidance includes detailed instructions of what options are available and how to configure each management functional capability listed.

Tests
The evaluator shall ensure that the MAS client can only access the applications specified for the group they are enrolled in. The evaluator shall create a user group, making sure that the MAS client user is excluded from the group. Verify that an application accessible to that group cannot

be accessed. The evaluator shall include the MAS client user in the group and assure that the application can be accessed.

FMT_SMF.1/SERVER_CONF_AGENT Specification of Management Functions (Server configuration of Agent)

FMT_SMF.1.1/SERVER_CONF_AGENT

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 or software
- 7. Query the current version of the hardware model of the device
- 8. Query the current version of installed 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: [**selection**:

- *13. Remove enterprise applications (MDF Function 17)*
- *14. Wipe enterprise data (MDF Function 28)*
- *15. Remove imported X.509v3 certificates and [**selection**:*
  - *no other X.509v3 certificates*
  - *[**assignment**: list of other categories of X.509v3 certificates]*

  ] in the Trust Anchor Database (MDF Function 12)
- *16. Alert the user*
- *17. Import keys or secrets into the secure key storage (MDF Function 9)*
- *18. Destroy imported keys or secrets and [**selection**:*
  - *no other keys or secrets*
  - *[**assignment**: list of other categories of keys or secrets]*

  ] in the secure key storage (MDF Function 10)
- *19. Read audit logs kept by the MD (MDF Function 32)*
- *20. Retrieve MD-software integrity verification values (MDF Function 38)*
- *21. Approve exceptions for sharing data between [**selection**:*
  - *application processes*
  - *group of application processes*

  ] (MDF Function 42)
- *22. Place applications into application process groups based on [**assignment**: application characteristics] (MDF Function 43)*
- *23. Revoke Biometric template (MDF Function 23)*
- *24. [**assignment**: list of other management functions to be provided by the MD]*
- *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 and disabled
  b. screen lock timeout
  c. number of authentication failures (MDF Function 2)
- 27. Wireless networks (SSIDs) to which the MD may connect (WLAN Client PP-Module Function 2)
- 28. Security policy for each wireless network:
  a. [**selection**:
     - *specify the CAs from which the MD will accept WLAN authentication server certificates*
     - *specify the FQDNs of acceptable WLAN authentication server certificates*

     ]
  b. ability to specify security type
  c. ability to specify authentication protocol
  d. specify the client credentials to be used for authentication

e. [**assignment**: *any additional WLAN management functions*] (WLAN Client PP-Module Function 1)

- 29. Application installation policy by [**selection**:
  - *specifying authorized application repositories*
  - *specifying a set of allowed applications and versions (an application allowlist)*
  - *denying application installation*

  ], (MDF Function 8)
- 30. Enable and disable policy for [**assignment**: *list of audio or visual collection devices*] across device, [**selection**:
  - *on a per-app basis*
  - *on a per-group of applications processes basis*
  - *no other method*

  ], (MDF Function 5)

and the following MD configuration policies :

[**selection**:

- *31. Enable and disable policy for the VPN protection across MD, [**selection**:*
  - *on a per-app basis*
  - *on a per-group of application processes basis*
  - *no other method*

  ] (MDF Function 3)
- *32. Enable and disable policy for [**assignment**: list of radios], (MDF Function 4)*
- *33. Enable and disable policy for data signaling over [**assignment**: list of externally accessible hardware ports], (MDF Function 24)*
- *34. Enable and disable policy for [**assignment**: list of protocols where the device acts as a server], (MDF Function 25)*
- *35. Enable and disable policy for developer modes, (MDF Function 26)*
- *36. Enable policy for data-at-rest protection, (MDF Function 20)*
- *37. Enable policy for removable media's data-at-rest protection, (MDF Function 21)*
- *38. Enable and disable policy for local authentication bypass, (MDF Function 27)*
- *39. The Bluetooth trusted channel policy:*
  a. *Disable/enable the Discoverable (for BR/EDR) and Advertising (for LE) modes (Bluetooth PP-Module Function 1)*
  b. *change the Bluetooth device name (Bluetooth PP-Module Function 2), [**selection**:*
     - *Provide separate controls for turning the BR/EDR and LE radios on and off (Bluetooth PP-Module Function 3)*
     - *allow and disallow additional wireless technologies to be used with Bluetooth (Bluetooth PP-Module Function 4)*
     - *configure allowable methods of Out of Band pairing (Bluetooth PP-Module Function 5)*
     - *disable and enable Advertising (for LE) (Bluetooth PP-Module Function 6)*
     - *disable and enable the Connection mode (Bluetooth PP-Module Function 7)*
     - *disable and enable the Bluetooth services or profiles available on the device (Bluetooth PP-Module Function 8)*
     - *specify minimum level of security for each pairing (Bluetooth PP-Module Function 9)*
     - *no other Bluetooth configuration*

     ]
- *40. Enable and disable policy for display notification in the locked state of*

  *[**selection**:*
  - *email notifications*
  - *calendar appointments*
  - *contact associated with phone call notification*
  - *text message notification*
  - *other application-based notifications*
  - *none*

  ] (MDF Function 18)
- *41. Policy for establishing a trusted channel or disallowing establishment if the MD cannot establish a connection to determine the validity of a certificate, (MDF Function 30)*
- *42. Enable and disable policy for the cellular protocols used to connect to cellular network base stations, (MDF Function 31)*

- *43. Policy for import and removal by applications of X.509v3 certificates in the Trust Anchor Database, (MDF Function 29)*
- *44. [**selection**:*
  - *certificate*
  - *public-key*

  ] used to validate digital signature on applications, (MDF Function 33)
- *45. Policy for exceptions for shared use of keys or secrets by multiple applications, (MDF Function 34)*
- *46. Policy for exceptions for destruction of keys or secrets by applications that did not import the key or secret, (MDF Function 35)*
- *47. The unlock banner policy, (MDF Function 36)*
- *48. Configure the auditable items (MDF Function 37)*
- *49. Enable and disable [**selection**:*
  - *USB mass storage mode*
  - *USB data transfer without user authentication*
  - *USB data transfer without authentication of the connection system*

  ] (MDF Function 39)
- *50. Enable and disable backup of [**selection**:*
  - *all applications*
  - *selected applications*
  - *selected groups of applications*
  - *configuration data*

  *] to [**selection**: locally connected system, remote system] (MDF Function 40)*
- *51. Enable and disable [**selection**:*
  - *Hotspot functionality authenticated by [**selection**: pre-shared key, passcode, no authentication]*
  - *USB tethering authenticated by [**selection**: pre-shared key, passcode, no authentication]*

  ] (MDF Function 41)
- *52. Enable and disable location services: [**selection**:*
  - *across device*
  - *on a per-app basis*
  - *on a per-group of application processes basis*
  - *no other method*

  ] (MDF Function 21)
- *53. Enable and disable policy for user unenrollment*
- *54. Enable and disable policy for the Always-On VPN protection across device (MDF Function 45)*
- *55. Enable and disable policy for use of Biometric Authentication Factor (MDF Function 22)*
- *56. Connectivity timeout policy: [**selection**:*
  - *allowed [**selection**: number of missed reachability events, length of time without server connectivity]*
  - *when server connectivity timeout is exceeded agent shall [**selection**: disable user password, wipe device] and [**selection**: [**assignment**: other action], none]*

  ]
- *57. Enable and disable multi-user modes*
- *58. Enable and disable automatic updates of system software*
- *59. Enable and disable removable media*
- *60. [**assignment**: list of other policies to be provided by the MD], no other policies*

].

Application Note: This requirement captures all the configuration functionality the TSF provides the administrator to configure the MDM agent. This requirement is broken into two configurable areas: MDM agent commands and MDM agent policies. The ST author can add more commands and configuration policies by completing the appropriate assignment statements.

Function-specific Application Notes:

Function-specific application notes reference Mobile Device Fundamentals (MDF) PP v4.0.

Function 12 may be satisfied for the BYOD use case by application denylisting or disabling. In the case of BYOD, an enterprise may not want to remove "personal" applications, thus for that use case disabling the application rather than

removing it would allow the user to not lose any information they might have in the application.

Function 16 provides the MDM server to display an alert to the user of the MD.

The audit data read according to Function 19 are to be transmitted to an external audit server according to FAU_STG.1. The MDM server is not expected to retain those logs.

Function 34 provides the ability to enable and disable policy for the list of protocols where the device acts as a server, such as a mobile hotspot.

Function 56 corresponds to FPT_NET_EXT.1.1 in agent. If the MDM agent has not had a successful reachability event with the MDM server in the amount of time specified in 'a', then the agent must perform the action selected in 'b'. It is feasible, but not required, that if multiple actions are selected in 'b', each action could be associated with a different timeout in 'a'. If function 56 is included in the ST, than FPT_NET_EXT.1.1 must be included in the agent ST.

Evaluation Activities

FMT_SMF.1/SERVER_CONF_AGENT
TSS
The evaluator shall examine the TSS to ensure that it describes each management function claimed. The evaluator shall examine the TSS to ensure that it identifies the management functions implemented for each supported MDM agent or platform, which are likely to be subsets of all of the management functions available to the administrator on the MDM server. The evaluator shall examine the TSS to verify that any differences between management functions and policies for each supported MDM agent or platform are clearly indicated.

The evaluator shall determine if the MD has been evaluated. If so, the evaluator shall examine the TSS to verify that it clearly identifies which management functions match the selections and assignments of the evaluated MD and which management functions were not evaluated.

Guidance
TBD

Tests
For each MDM agent or platform listed as supported in the ST:

- *Test FMT_SMF.1/SERVER_CONF_AGENT:1: The evaluator shall verify the ability to command each MDM agent functional capability and configure each MDM agent policy listed above.*

FMT_SMF.1/SERVER_CONF_SERVER Specification of Management Functions (Server Configuration of Server)

FMT_SMF.1.1/SERVER_CONF_SERVER

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

1. Configure the [**selection**:
   - *devices specified by [**selection**: IMEI, [**assignment**: a unique device ID]]*
   - *specific device models*
   - *a number of devices*
   - *specific time period*

   ] and [**selection**: *[**assignment**: other features], no other features*] allowed for enrollment
2. [**selection**: *1. Choose X.509v3 certificates for MDM server use, 2. Allow the administrator to choose whether to accept the certificate when connection cannot be made to establish validity, 3. Configure the TOE unlock banner, 4. Configure periodicity of the following commands to the agent: [**assignment**: list of commands], 5. Configure the privacy-sensitive information that will and will not be collected from particular MDs, 6. Configure the length of time the enrollment authenticator is valid, 7. Configure the interaction between TOE components, 8. Configure the cryptographic functionality, 9. [**assignment**: other management functions], 10. No other management functions*]

Application Note: This requirement captures all the configuration functionality in the MDM server to configure the underlying MDM server. The ST author can

add more commands and configuration policies by completing the assignment statement.

Function b1 is selection-based under the very limited circumstances when certificate management is both performed solely by platform-provided functionality and the platform uses cloud services; for all other instances it is mandatory. This requirement will be made fully mandatory again in a future version of the PP. Function b1 can be met by relying on the platform to configure the certificates used by the MDM server, however, the MDM server must allow the administrator to choose which certificate is used for a specific functionality. The selection in a corresponds to the selection in FIA_ENR_EXT.1.2. The selection in c.2 includes a function that corresponds to the selection in FIA_X509_EXT.2.2. Function c.4 allows the administrator to configure periodicity of assigned commands, for example "read audit logs kept by the MD." In this way the administrator can configure the MDM system to retrieve audit logs from the MD on a periodic, such as daily, basis in order to ensure freshness of log data and to minimize loss of audit logs. Function c.5 allows the administrator to configure the privacy-sensitive information that will and will not be collected from particular MDs to handle BYOD environments where some information may not be appropriate to collect. Privacy sensitive information may include items such as device physical location and lists of installed personal applications, and would vary depending on the particular capabilities of the TOE and MDM agent. The TOE should provide the capability to group enrolled devices into categories such as enterprise-owned and personal-owned and define the information that will be collected from devices in each category. Function c.6 corresponds to configuring the length of time the user authenticator for enrollment is valid in FMT_SAE_EXT.1. This function must be included in the ST if and only if FMT_SAE_EXT.1 is included in the ST.

For distributed TOEs, the interaction between TOE components will be configurable (see FCO_CPC_EXT.1). Therefore, the ST author includes the selection "Ability to configure the interaction between TOE components" for distributed TOEs. A simple example would be the change of communication protocol according to FPT_ITT.1/INTER_XFER. Another example would be changing the management of a TOE component from direct remote administration to remote administration through another TOE component. A more complex use case would be if the realization of an SFR is achieved through two or more TOE components and the responsibilities between the two or more components could be modified.

For distributed TOEs, that implement a registration channel (as described in FCO_CPC_EXT.1.2), the ST author uses the selection "configure the cryptographic functionality" in this SFR, and its corresponding mapping in the TSS, to describe the configuration of any cryptographic aspects of the registration channel that can be modified by the operational environment in order to improve the channel security.

Evaluation Activities

FMT_SMF.1/SERVER_CONF_SERVER
TSS
The evaluator shall examine the TSS to ensure that it describes each management function listed. For function c.5, the evaluator shall examine the TSS to ensure that it describes the privacy-sensitive information that the TOE has the capability to collect from enrolled MDs.

Guidance
The evaluator shall verify the operational guidance includes detailed instructions of what options are available and how to configure each management functional capability listed.

Tests
The tests of functions a, c.2, c.3, and c.6 are performed in conjunction with the use of the function. Test 3 also covers function c.5. The evaluator shall perform the following tests:

- *Test FMT_SMF.1/SERVER_CONF_SERVER:1: The evaluator shall configure the TSF authentication certificates and verify that the correct certificate is used in established trusted connections (FPT_ITT.1/INTER_XFER, FPT_ITT.1/INTER_XFER_AGENT, FTP_ITC.1/INTER_XFER_IT, and FTP_TRP.1/TRUSTPATH_ENROLL).*
- *Test FMT_SMF.1/SERVER_CONF_SERVER:2: (conditional) The evaluator shall configure the periodicity for the assigned list of commands to the agent for several configured time periods and shall verify that the MDM server performs the commands schedule.*
- *Test FMT_SMF.1/SERVER_CONF_SERVER:3: (conditional) The evaluator shall design and perform tests to demonstrate that the assigned function may be configured and that the intended behavior of the function is enacted by the MDM server.*

FMT_SMR.1/SECMAN_ROLES Security Roles

FMT_SMR.1.1/SECMAN_ROLES

The TSF shall maintain the roles **administrator, MD user, and**[**selection**: ***[assignment: additional authorized identified roles]*** , ***no additional roles***].

FMT_SMR.1.2/SECMAN_ROLES

The TSF shall be able to associate users with roles.

Application Note: It is envisioned that the MDM server will be configured and maintained by different user roles. The assignment is used by the ST author to list the roles that are supported. At a minimum, one administrative role must be supported. If no additional roles are supported, then "no additional roles" is selected. The MD user role is used for enrollment of MDs to the MDM according to FIA_ENR_EXT.1.

For distributed TOEs, not every TOE component is required to implement its own user management to fulfill this SFR. At least one component has to support authentication and identification of users according to FIA_UAU.1. For the other TOE components authentication can be realized through the use of a trusted channel (either according to FTP_ITC.1 or FPT_ITT.1/INTER_XFER / FPT_ITT.1/INTER_XFER_AGENT) from a component that supports the authentication of users according to FIA_UAU.1. The identification of users according to FIA_UAU.1.2 and the association of users with roles according to FMT_SMR.1.2/SECMAN_ROLES is done through the components that support the authentication of users according to FIA_UAU.1.

Evaluation Activities

FMT_SMR.1/SECMAN_ROLES
TSS
The evaluator shall examine the TSS to verify that it describes the administrator role and the powers granted to and limitations of the role.

Guidance
The evaluator shall review the operational guidance to ensure that it contains instructions for administering the TOE and which interfaces are supported.

Tests
In the course of performing the testing activities for the evaluation, the evaluator shall use all supported interfaces, although it is not necessary to repeat each test involving an administrative action with each interface. The evaluator shall ensure, however, that each supported method of administering the TOE that conforms to the requirements of this PP be tested; for instance, if the TOE can be administered through a local hardware interface or HTTPS then both methods of administration must be exercised during the evaluation team's test activities.

FMT_SMR.1/SECMAN_ROLES_MAS Security Roles (MAS Server)

This is a selection-based component. Its inclusion depends upon selection from FMT_MOF.1.1/FUNCBE.

FMT_SMR.1.1/SECMAN_ROLES_MAS

The TSF shall **additionally** maintain the roles of [*enrolled managed devices, application access groups, and [**assignment**: additional authorized identified roles]*].

FMT_SMR.1.2/SECMAN_ROLES_MAS

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

Application Note: It is envisioned that the MAS Server will be configured and maintained by different user roles. The assignment is used by the ST author to list the roles that are supported. At a minimum, one administrative role must be supported. If no additional roles are supported, then "no additional roles" is stated. The MD user role is used for enrollment of MDs to the MAS according to FIA_ENR_EXT.1.

This requirement is claimed if "enable, disable, and modify policies listed in FMT_SMF.1/MAS" is selected in FMT_MOF.1.1/FUNCBE.

Evaluation Activities

FMT_SMR.1/SECMAN_ROLES_MAS
TSS
The evaluator shall examine the TSS to verify that it describes the administrator role and the powers granted to and limitations of the role.

Guidance
The evaluator shall review the operational guidance to ensure that it contains instructions for administering the TOE and which interfaces are supported.

Tests
In the course of performing the testing activities for the evaluation, the evaluator shall use all supported interfaces, although it is not necessary to repeat each test involving an administrative action with each interface. The evaluator shall ensure, however, that each supported method of administering the TOE that conforms to the requirements of this PP be tested; for instance, if the TOE can be administered through a local hardware interface or HTTPS then both methods of administration must be exercised during the evaluation team's test activities.

4.1.7 Protection of the TSF (FPT)

FPT_API_EXT.1 Use of Supported Services and APIs

FPT_API_EXT.1.1

The TSF shall use only documented platform APIs.

Application Note: This requirement applies to the APIs used when "invoke platform-provided functionality" is selected in an SFR. The definition of documented may vary depending on whether the MDM software is provided by a third-party (who relies on documented platform APIs) or by a platform vendor who may be able to guarantee support for platform APIs.

Evaluation Activities

FPT_API_EXT.1
TSS
The evaluator shall verify that the TSS lists the platform APIs used by the MDM software. The evaluator shall then compare the list with the supported APIs (available through e.g., developer accounts, platform developer groups) and ensure that all APIs listed in the TSS are supported.

Guidance
TBD

Tests
TBD

FPT_FLS.1 Failure with Preservation of Secure State

FPT_FLS.1.1

The TSF shall preserve a secure state when the following types of failures occur: [*DRBG self-test failure*].

Application Note: The intent of this requirement is to ensure that cryptographic services requiring random bit generation cannot be performed if a failure of a self-test defined in FPT_TST.1 occurs.

Evaluation Activities

FPT_FLS.1
TSS
The evaluator shall verify that the TSF describes how the TOE enters an error state in the event of a DRBG self-test failure.

Guidance
The evaluator shall verify that the guidance documentation describes the error state that results from a DRBG self-test failure and the actions that a user or administrator should take in response to attempt to resolve the error state.

Tests
There are no test activities for this component.

FPT_ITT.1/INTER_XFER Internal TOE TSF Data Transfer

This is a selection-based component. Its inclusion depends upon selection from FTP_ITC_EXT.1.1.

FPT_ITT.1.1/INTER_XFER

The TSF shall [**selection**:

- ***invoke platform-provided functionality to use [selection:***
  - ***IPsec***
  - ***mutually authenticated TLS***
  - ***mutually authenticated DTLS***
  - ***HTTPS***
  - ***SSH***

  ]
- ***implement functionality using [selection:***
  - ***IPsec as defined in the PP-Module for VPN Client***
  - ***mutually authenticated TLS as defined in the Package for Transport Layer Security***
  - ***mutually authenticated DTLS as defined in the Package for Transport Layer Security***
  - ***HTTPS in accordance with FCS_HTTPS_EXT.1***
  - ***SSH as defined in the Functional Package for Secure Shell***

  ]

] **to** protect **all** data from [*disclosure and modification*] when it is **transferred** between separate parts of the TOE.

Application Note: This requirement ensures all communications between components of a distributed TOE are protected through the use of an encrypted communications channel. The data passed in this trusted communication channel are encrypted as defined in the protocol chosen in the second selection.

The trusted channel uses secure protocols that preserve the confidentiality and integrity of MDM communications. The ST author chooses the mechanism or mechanisms supported by the TOE. To support mutual authentication FIA_X509_EXT.1/CERTVAL_SEL should be included in the ST. This channel may also be used as the registration channel for the registration process, as described in section 3.1 and FCO_CPC_EXT.1.2.

If "IPsec as defined in the PP-Module for VPN Client" is selected, the TSF must claim conformance to a PP-Configuration that includes the VPN Client PP-Module.

If the ST author selects "SSH as defined in the Functional Package for Secure Shell," the TSF must be validated against the FP for Secure Shell with the MDM PP. It should be noted that due to constraints imposed by this PP that SHA-1 cannot be used.

If the ST author selects "mutually authenticated TLS as defined in the Package for Transport Layer Security" or "mutually authenticated DTLS as defined in the Package for Transport Layer Security," the TSF must be validated against requirements from the Package for Transport Layer Security, with the following selections made:

- FCS_TLS_EXT.1:
  - either TLS or DTLS is selected depending on the selection made in
  - either client or server is selected as appropriate
- FCS_TLSC_EXT.1.1 or FCS_TLSS_EXT.1.1 (as appropriate):
  - The cipher suites selected must correspond with the algorithms and hash functions allowed in FCS_COP.1.
  - mutual authentication must be selected

Protocol, RBG, Certificate validation, algorithm, and similar services may be met with platform-provided services.

This requirement is claimed if "FPT_ITT.1/INTER_XFER" is selected in FTP_ITC_EXT.1.

If HTTPS is chosen, FCS_HTTPS_EXT.1 must be included in the ST.

Evaluation Activities

FPT_ITT.1/INTER_XFER
TSS
The evaluator shall examine the TSS to determine that the methods and protocols used to protect distributed TOE components are described. The evaluator shall also confirm that all protocols listed in the TSS in support of TOE administration are consistent with those specified in the requirement, and are included in the requirements in the ST.

If "invoke platform-provided functionality" is selected, the evaluator shall examine the TSS to verify that it describes (for each supported platform) how this functionality is invoked (it should be noted that this may be through a mechanism that is not implemented by the MDM server; nonetheless, that mechanism will be identified in the TSS as part of this evaluation activity).

Guidance
The evaluator shall confirm that the operational guidance contains instructions for establishing the communication paths for each supported method.

Tests
- *Test FPT_ITT.1/INTER_XFER:1: The evaluator shall ensure that communications using each specified (in the operational guidance) communication method is tested during the course of the evaluation, setting up the connections as described in the operational guidance and ensuring that communication is successful.*
- *Test FPT_ITT.1/INTER_XFER:2: The evaluator shall ensure, for each method of communication, the channel data is not sent in plaintext.*

Further evaluation activities are associated with the specific protocols.

FPT_ITT.1/INTER_XFER_AGENT Internal TOE TSF Data Transfer (MDM Agent)

This is a selection-based component. Its inclusion depends upon selection from FTP_ITC_EXT.1.1.

FPT_ITT.1.1/INTER_XFER_AGENT

The TSF shall [**selection**:
- ***invoke platform-provided functionality to use [selection:***
  - ***mutually authenticated TLS***
  - ***mutually authenticated DTLS***
  - ***HTTPS***

  ]
- ***implement functionality using [selection:***
  - ***mutually authenticated TLS as defined in the Package for Transport Layer Security***
  - ***mutually authenticated DTLS 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: This requirement ensures all communications between the TSF and MDM agent are protected through the use of an encrypted communications channel. The data passed in this trusted communication channel are encrypted as defined in the protocol chosen in the second selection.

The trusted channel uses secure protocols that preserve the confidentiality and integrity of MDM communications. The ST author chooses the mechanism or mechanisms supported by the TOE. To support mutual authentication FIA_X509_EXT.1/CERTVAL_MAN should be included in the ST. This channel may also be used as the registration channel for the registration process, as described in section 3.1 and FCO_CPC_EXT.1.2.

If the ST author selects "mutually authenticated TLS as defined in the Package for Transport Layer Security" or "mutually authenticated DTLS as defined in the Package for Transport Layer Security," the TSF must be validated against

requirements from the Package for Transport Layer Security, with the following selections made:

- FCS_TLS_EXT.1:
  - either TLS or DTLS is selected depending on the selection made in FPT_ITT.1.1(2)
  - either client or server is selected as appropriate
- FCS_TLSC_EXT.1.1 or FCS_TLSS_EXT.1.1 (as appropriate):
  - The cipher suites selected must correspond with the algorithms and hash functions allowed in FCS_COP.1.
  - mutual authentication must be selected

Protocol, RBG, Certificate validation, algorithm, and similar services may be met with platform-provided services.

If HTTPS is chosen, you must include FCS_HTTPS_EXT.1 in the ST. This requirement is claimed if "FPT_ITT.1/INTER_XFER_AGENT" is selected in FTP_ITC_EXT.1.

If HTTPS is chosen, FCS_HTTPS_EXT.1 must be included in the ST.

Evaluation Activities

FPT_ITT.1/INTER_XFER_AGENT
TSS
The evaluator shall examine the TSS to determine that the methods and protocols used to protect distributed TOE components are described. The evaluator shall also confirm that all protocols listed in the TSS in support of TOE administration are consistent with those specified in the requirement, and are included in the requirements in the ST.

If "invoke platform-provided functionality" is selected, the evaluator shall examine the TSS to verify that it describes (for each supported platform) how this functionality is invoked (it should be noted that this may be through a mechanism that is not implemented by the MDM server; nonetheless, that mechanism will be identified in the TSS as part of this evaluation activity).

Guidance
The evaluator shall confirm that the operational guidance contains instructions for establishing the communication paths for each supported method.

Tests

- *Test FPT_ITT.1/INTER_XFER_AGENT:1: The evaluator shall ensure that communications using each specified (in the operational guidance) communication method is tested during the course of the evaluation, setting up the connections as described in the operational guidance and ensuring that communication is successful.*
- *Test FPT_ITT.1/INTER_XFER_AGENT:2: The evaluator shall ensure, for each method of communication, the channel data is not sent in plaintext.*

Further evaluation activities are associated with the specific protocols.

FPT_LIB_EXT.1 Use of Third-Party Libraries

FPT_LIB_EXT.1.1

The MDM software shall be packaged with only [**assignment**: *list of third-party libraries*].

Application Note: The intention of this requirement is for the ST author to document which software libraries (including version numbers) the MDM software is including in case vulnerabilities are later discovered with those libraries. For cloud services, the assignment can be completed with a reference to a non-public library document as an annex for those libraries that are not or cannot be made public.

Evaluation Activities

FPT_LIB_EXT.1
TSS
The evaluator shall verify that the TSS lists the libraries used by the MDM software or the public libraries used by the cloud MDM services. The evaluator shall verify that libraries found to be packaged with or employed by the MDM software are limited to those in the assignment.

Library Document

(conditional: The TOE is a cloud MDM service) The evaluator shall verify that the non-public library document lists the libraries used by the MDM service.

Guidance
TBD

Tests
TBD

FPT_TST.1 TSF Self-Testing

FPT_TST.1.1

The TSF shall run a suite of the following self-tests [**selection**: *during initial start-up, periodically during normal operation, at the request of the authorized user, at the conditions [**assignment**: conditions under which self-test should occur]*] to demonstrate the correct operation of [*TSF DRBG specified in FCS_RBG.1*].

FPT_TST.1.2

The TSF shall provide authorized users with the capability to verify the integrity of [*[DRBG seed and output data]*].

FPT_TST.1.3

The TSF shall provide authorized users with the capability to verify the integrity of [*[TSF DRBG specified in FCS_RBG.1]*].

Application Note: This SFR is a required dependency of FCS_RBG.1. It is intended to require that any DRBG implemented by the TOE undergo health testing to ensure that the random bit generation functionality has not been degraded. If the TSF supports multiple DRBGs, this SFR should be iterated to describe the self-test behavior for each.

Evaluation Activities

FPT_TST.1
TSS
The evaluator shall examine the TSS to ensure that it details the self-tests that are run by the TSF along with how they are run. This description should include an outline of what the tests are actually doing. The evaluator shall ensure that the TSS makes an argument that the tests are sufficient to demonstrate that the DRBG is operating correctly.

Note that this information may also be placed in the entropy documentation specified by Appendix D - Entropy Documentation and Assessment.

Guidance
If a self-test can be executed at the request of an authorized user, the evaluator shall verify that the operational guidance provides instructions on how to execute that self-test.

Tests
For each self-test, the evaluator shall verify that evidence is produced that the self-test is executed when specified by FPT_TST.1.1.

If a self-test can be executed at the request of an authorized user, the evaluator shall verify that following the steps documented in the operational guidance to perform the self-test will result in execution of the self-test.

FPT_TST_EXT.1 Functionality Testing

FPT_TST_EXT.1.1

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

FPT_TST_EXT.1.2

The TSF shall [**selection**: *invoke platform-provided functionality, 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 [**selection**: *TSF, TOE platform*] -provided cryptographic services.

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

For distributed TOEs, all TOE components (except the Agent components) have to perform self-tests. This does not necessarily mean that each TOE component has to carry out the same self-tests: the ST describes the applicability of the selection (i.e., when self-tests are run) and the final assignment (i.e., which self-tests are carried out) to each TOE component.

Evaluation Activities

FPT_TST_EXT.1
TSS
If "invoke platform-provided functionality" is selected, the evaluator shall examine the TSS to verify that it describes (for each supported platform) how this functionality is invoked (it should be noted that this may be through a mechanism that is not implemented by the MDM server; nonetheless, that mechanism will be identified in the TSS as part of this evaluation activity).

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

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

Guidance
TBD

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

FPT_TUD_EXT.1 Trusted Update

FPT_TUD_EXT.1.1

The TSF shall provide authorized administrators the ability to query the current version of the software.

Application Note: For a distributed TOE, the method of determining the installed versions on each component of the TOE is described in the operational guidance. In the requirement, "software" refers to the component of the distributed TOE to which the requirement is being applied.

FPT_TUD_EXT.1.2

The TSF shall [**selection**: *invoke platform-provided functionality*, *implement functionality*] to provide authorized administrators the ability to initiate updates to TSF software.

FPT_TUD_EXT.1.3

The TSF shall [**selection**: *invoke platform-provided functionality*, *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: The software on the TSF will occasionally need to be updated. This requirement is intended to ensure that the TSF only installs updates provided by the vendor, as updates provided by another source may

contain malicious code. If the server is not an appliance, the update will be verified by the platform on which the server software runs. If the server is an appliance, the update must be verified by the TSF software or hardware.

For distributed TOEs, all TOE components must support Trusted Update. The verification of the signature or hash on the update must either be done by each TOE component itself (signature verification) or for each TOE component (hash verification).

Updating a distributed TOE might lead to the situation where different TOE components are running different software versions. Depending on the differences between the different software versions the impact of a mixture of different software versions might be no problem at all or critical to the proper functioning of the TOE. The TSS must detail the mechanisms that support the continuous proper functioning of the TOE during trusted update of distributed TOEs.

Evaluation Activities

FPT_TUD_EXT.1.1
TSS
TBD

Guidance
The evaluator shall ensure that the administrator guidance includes instructions for determining the current version of the TOE.

Tests
The evaluator shall query the TSF for the current version of the software according to the operational guidance and shall verify that the current version matches that of the documented and installed version.

FPT_TUD_EXT.1.2
TSS
If "invoke platform-provided functionality" is selected, the evaluator shall examine the TSS to verify that it describes (for each supported platform) how this functionality is invoked (it should be noted that this may be through a mechanism that is not implemented by the MDM server; nonetheless, that mechanism will be identified in the TSS as part of this evaluation activity).

Guidance
TBD

Tests
TBD

FPT_TUD_EXT.1.3
TSS
If "invoke platform-provided functionality" is selected, the evaluator shall examine the TSS to verify that it describes (for each supported platform) how this functionality is invoked (it should be noted that this may be through a mechanism that is not implemented by the MDM server; nonetheless, that mechanism will be identified in the TSS as part of this evaluation activity).

If "implement functionality" is selected, the evaluator shall examine the TSS and verify that it describes the standards by which the updates are digitally signed and how the signature verification process is implemented.

Guidance
The evaluator shall examine the operational guidance to verify that it describes how to query the current version of the TSF software, how to initiate updates and how to check the integrity of updates prior to installation.

Tests
The evaluator shall perform the following tests:
- *Test FPT_TUD_EXT.1.3:1: The evaluator shall attempt to initiate an update digitally signed by the vendor and verify that the update is successfully installed.*
- *Test FPT_TUD_EXT.1.3:2: The evaluator shall attempt to install an update not digitally signed by the vendor and verify that either the signature can be checked (allowing the update to be aborted) or the update is not installed.*

4.1.8 TOE Access (FTA)

FTA_TAB.1 Default TOE Access Banners

This is an optional component. However, applied modules or packages might redefine it

as mandatory.

FTA_TAB.1.1

Before establishing a user session, the [*TSF*] shall [**selection**: ***invoke platform-provided functionality***, ***implement functionality***] **to** display an [*administrator-specified advisory notice and consent warning*] message **regarding use of the TOE.**

Application Note: This requirement is to ensure that an advisory notice or consent banner is presented to the user on start-up or unlock of the TSF.

Evaluation Activities

FTA_TAB.1
TSS
If "invoke platform-provided functionality" is selected, the evaluator shall examine the TSS to verify that it describes (for each supported platform) how this functionality is invoked (it should be noted that this may be through a mechanism that is not implemented by the MDM server; nonetheless, that mechanism will be identified in the TSS as part of this evaluation activity).

If "implement functionality" is selected, the TSS shall describe when the banner is displayed.

Guidance
The evaluator follows the operational guidance to configure a notice and consent warning message.

Tests
The evaluator shall also perform the following test: The evaluator shall start up or unlock the TSF. The evaluator shall verify that the notice and consent warning message is displayed in each instance described in the TSS.

4.1.9 Trusted Path/Channels (FTP)

FTP_ITC.1/INTER_TSF_XFER_AGENT Inter-TSF Trusted Channel (MDM agent)

This is a selection-based component. Its inclusion depends upon selection from FTP_ITC_EXT.1.1.

FTP_ITC.1.1/INTER_TSF_XFER_AGENT

The TSF shall [**selection**:

- ***invoke platform-provided functionality to use [selection:***
  - ***mutually authenticated TLS***
  - ***mutually authenticated DTLS***
  - ***HTTPS***

  ]
- ***implement functionality using [selection:***
  - ***mutually authenticated TLS as defined in the Package for Transport Layer Security***
  - ***mutually authenticated DTLS as defined in the Package for Transport Layer Security***
  - ***HTTPS in accordance with FCS_HTTPS_EXT.1***

  ]

] **to** provide a **trusted** communication channel between itself **(as a server) and the MDM agent** that is logically distinct from other communication channels, provides assured identification of its endpoints, **protects channel data from disclosure, and detects modification of the channel data.**

Application Note: The intent of the mandatory portion of the above requirement is to use the cryptographic protocols identified in the requirement to establish and maintain a trusted channel between the TOE and the MDM agent. If the TOE includes a separate MAS Server, this requirement also addresses the communication between the MAS Server and the MDM agent. Only TLS, DTLS, or HTTPS are used in this trusted channel.

This requirement is to ensure that the transmission of any audit logs, MD information data (software version, hardware model, and application versions), and configuration data collected by the MDM agent and sent from the MDM

agent to the MDM Server, when commanded, or at configurable intervals, is properly protected. This trusted channel also protects any commands and policies sent by the MDM server to the MDM agent. Either the MDM agent or the MDM server is able to initiate the connection.

For TLS connections between the MDM server and agent, the MDM server is the Server side of the TLS connection, therefore it is appropriate to include the selection-based FCS_TLSS SFRs in the ST, not FCS_TLSC SFRs. With respect to mutual authentication, in cases where the agent is outside of the TOE, it should be verified that the server can support mutual authentication, meaning that the server includes support for client-side certificates for TLS mutual authentication post-enrollment. However, the client side is not evaluated since the agent is not in the TOE.

This trusted channel protects the connection between an enrolled MDM agent and the MDM server. FTP_TRP.1/TRUSTPATH_ENROLL provides a trusted channel to protect the connection between an unenrolled MDM agent and the MDM server during the enrollment operation.

The trusted channel uses TLS, DTLS, or HTTPS as the protocol that preserves the confidentiality and integrity of MDM communications. The ST author chooses the mechanism or mechanisms supported by the TOE.

If the ST author selects "mutually authenticated TLS as defined in the Package for Transport Layer Security" or "mutually authenticated DTLS as defined in the Package for Transport Layer Security," the TSF must be validated against requirements from the Package for Transport Layer Security, with the following selections made:

- FCS_TLS_EXT.1:
  - either TLS or DTLS is selected depending on the selection made in FTP_ITC.1.1/INTER_TSF_XFER_AGENT
  - server must be selected
- FCS_TLSS_EXT.1.1:
  - The cipher suites selected must correspond with the algorithms and hash functions allowed in FCS_COP.1.
  - mutual authentication must be selected

Protocol, RBG, Certificate validation, algorithm, and similar services may be met with platform-provided services.

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

This requirement is claimed if "FTP_ITC.1/INTER_TSF_XFER_AGENT" is selected in FTP_ITC_EXT.1.

If HTTPS is chosen, FCS_HTTPS_EXT.1 must be included in the ST.

FTP_ITC.1.2/INTER_TSF_XFER_AGENT

The TSF shall **[selection: *invoke platform-provided functionality*, *implement functionality*] to** permit the [*TSF and MDM agent*] to initiate communication via the trusted channel.

FTP_ITC.1.3/INTER_TSF_XFER_AGENT

The TSF shall **[selection: *invoke platform-provided functionality*, *implement functionality*] to** initiate communication via the trusted channel for [*all communication between the TSF and the MDM agent*]

Evaluation Activities

FTP_ITC.1/INTER_TSF_XFER_AGENT
TSS
The evaluator shall examine the TSS to determine that the methods of agent-server communication are indicated, along with how those communications are protected. The evaluator shall also confirm that all protocols listed in the TSS in support of remote TOE administration are consistent with those specified in the requirement, and are included in the requirements in the ST.

If "invoke platform-provided functionality" is selected, the evaluator shall examine the TSS to verify that it describes (for each supported platform) how this functionality is invoked (it should be noted that this may be through a mechanism that is not implemented by the MDM server; nonetheless, that mechanism will be identified in the TSS as part of this evaluation activity).

Guidance
The evaluator shall confirm that the operational guidance contains instructions for configuring the communication channel between the MDM agent and the MDM server for each supported method.

Tests

- *Test FTP_ITC.1/INTER_TSF_XFER_AGENT:1: The evaluator shall ensure that communications using each specified (in the operational guidance) agent-server communication method is tested during the course of the evaluation, setting up the connections as described in the operational guidance and ensuring that communication is successful.*
- *Test FTP_ITC.1/INTER_TSF_XFER_AGENT:2: The evaluator shall ensure, for each method of agent-server communication, the channel data is not sent in plaintext.*
- *Test FTP_ITC.1/INTER_TSF_XFER_AGENT:3: The evaluator shall ensure, for each communication channel with the MDM server, that a protocol analyzer identifies the traffic as the protocol under testing.*

Further evaluation activities are associated with the specific protocols.

FTP_ITC.1/INTER_XFER_IT Inter-TSF Trusted Channel (Authorized IT Entities)

FTP_ITC.1.1/INTER_XFER_IT

The TSF shall [**selection**:

- ***invoke platform-provided functionality to use [selection:***
  - ***IPsec***
  - ***SSH***
  - ***mutually authenticated TLS***
  - ***mutually authenticated DTLS***
  - ***HTTPS***

  ]
- ***implement functionality using [selection:***
  - ***IPsec as defined in the PP-Module for VPN Client***
  - ***SSH as defined in the Functional Package for Secure Shell***
  - ***mutually authenticated TLS as defined in the Package for Transport Layer Security***
  - ***mutually authenticated DTLS as defined in the Package for Transport Layer Security***
  - ***HTTPS in accordance with FCS_HTTPS_EXT.1***

  ]

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

Application Note: The intent of the mandatory portion of the above requirement is to use the cryptographic protocols identified in the requirement to establish and maintain a trusted channel with authorized IT entities that the TOE interacts with to perform its functions.

Protection (by one of the listed protocols) is required at least for communications with the server that collects the audit information. If it communicates with an authentication server (e.g., RADIUS), then the ST author chooses "authentication server" in FTP_ITC.1.1/INTER_XFER_IT and this connection must be protected by one of the listed protocols. If other authorized IT entities (e.g., NTP server) are protected, the ST author makes the appropriate assignments (for those entities) and selections (for the protocols that are used to protect those connections).

To summarize, the connection to an external audit collection server is required to be protected by one of the listed protocols. If an external authentication server is supported, then it is required to protect that connection with one of the listed protocols. For any other external server, external communications are not required to be protected, but if protection is claimed, then it must be protected

with one of the identified protocols.

For communications with any authorized IT entities outside of the TOE, the MDM server should still be able to support mutual authentication. There are no requirements levied on the external entities, but the MDM server should be able to support mutual authentication. This way if the non-TOE authorized entity does support mutual authentication, the MDM server is in a position to make use of that.

The trusted channel uses IPsec, TLS, DTLS, or HTTPS as the protocol that preserves the confidentiality and integrity of MDM communications. The ST author chooses the mechanism or mechanisms supported by the TOE.

If "IPsec as defined in the PP-Module for VPN Client" is selected, the TSF must claim conformance to a PP-Configuration that includes the VPN Client PP-Module.

If HTTPS is chosen, FCS_HTTPS_EXT.1 must be included in the ST.

If the ST author selects "SSH as defined in the Functional Package for Secure Shell," the TSF must be validated against the FP for Secure Shell with the MDM PP. It should be noted that due to constraints imposed by this PP that SHA-1 cannot be used.

If the ST author selects "mutually authenticated TLS as defined in the Package for Transport Layer Security" or "mutually authenticated DTLS as defined in the Package for Transport Layer Security," the TSF must be validated against requirements from the Package for Transport Layer Security, with the following selections made:

- FCS_TLS_EXT.1:
  - either TLS or DTLS is selected depending on the selection made in FTP_ITC.1.1/INTER_XFER_IT
  - either client or server is selected as appropriate
- FCS_TLSC_EXT.1.1 or FCS_TLSS_EXT.1.1 (as appropriate):
  - The cipher suites selected must correspond with the algorithms and hash functions allowed in FCS_COP.1.
  - mutual authentication must be selected
- FCS_DTLSC_EXT.1.1 or FCS_DTLSS_EXT.1.1 (as appropriate):
  - The cipher suites selected must correspond with the algorithms and hash functions allowed in FCS_COP.1.
  - mutual authentication must be selected

Protocol, RBG, Certificate validation, algorithm, and similar services may be met with platform-provided services.

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

FTP_ITC.1.2/INTER_XFER_IT

The TSF shall [**selection**: ***invoke platform-provided functionality***, ***implement functionality***] **to** permit **the MDM server or other authorized IT entities** to initiate communication via the trusted channel.

FTP_ITC.1.3/INTER_XFER_IT

The TSF shall [**selection**: ***invoke platform-provided functionality*** , ***implement functionality***] **to** initiate communication via the trusted channel for [**assignment**: *list of functions for which a trusted channel is required*].

Application Note: While there are no requirements on the party initiating the communication, the ST author lists in the assignment for FTP_ITC.1.3/INTER_XFER_IT the services for which the TOE can initiate the communication with the authorized IT entity.

Evaluation Activities

FTP_ITC.1/INTER_XFER_IT

TSS
The evaluator shall examine the TSS to determine that the methods of communication with authorized IT entities are indicated, along with how those communications are protected.

If "invoke platform-provided functionality" is selected, the evaluator shall examine the TSS to verify that it describes (for each supported platform) how this functionality is invoked (it should be noted that this may be through a mechanism that is not implemented by the MDM server; nonetheless, that mechanism will be identified in the TSS as part of this evaluation activity).

Guidance
The evaluator shall confirm that the operational guidance contains instructions for configuring the communication channel between the MDM server and authorized IT entities for each supported method.

Tests
- *Test FTP_ITC.1/INTER_XFER_IT:1: The evaluator shall ensure that communications using each specified (in the operational guidance) communication method is tested during the course of the evaluation, setting up the connections as described in the operational guidance and ensuring that communication is successful.*
- *Test FTP_ITC.1/INTER_XFER_IT:2: The evaluator shall ensure, for each method of communication, the channel data is not sent in plaintext.*
- *Test FTP_ITC.1/INTER_XFER_IT:3: The evaluator shall ensure, for each communication channel with the MDM server, that a protocol analyzer identifies the traffic as the protocol under testing.*

Further evaluation activities are associated with the specific protocols.

FTP_ITC_EXT.1 Trusted Channel

FTP_ITC_EXT.1.1

The TSF shall provide a communication channel between itself and [**selection**:
- *an MDM agent that is internal to the TOE*
- *an MDM agent that is external to the TOE*
- *other components comprising the distributed TOE*

] that is logically distinct from other communication channels, as specified in [**selection**: *FPT_ITT.1/INTER_XFER*, *FPT_ITT.1/INTER_XFER_AGENT*, *FTP_ITC.1/INTER_TSF_XFER_AGENT*].

Application Note: This SFR describes the communication channels between the MDM server and an agent that may or may not be in the TOE, and optionally separate components comprising a distributed TOE, to determine which selection-based SFRs will need to be included in the ST.

If the TOE includes an MDM agent, then the communication channel between the agent and TSF is internal to the TOE and thus "an MDM agent that is internal to the TOE" must be selected in the first selection. Additionally FPT_ITT.1/INTER_XFER_AGENT must be selected in the second selection and FPT_ITT.1/INTER_XFER_AGENT must be included in the ST.

If the TOE interoperates with an evaluated MDM agent built into a MD (i.e., the TOE does not include an MDM agent), the communication channel between the agent and TSF is external to the TOE and thus "an MDM agent that is external to the TOE" must be selected in the first selection. Additionally FTP_ITC.1/INTER_TSF_XFER_AGENT must be selected in the second selection and FTP_ITC.1/INTER_TSF_XFER_AGENT must be included in the ST.

If "other components comprising the distributed TOE" is selected in the first selection then FPT_ITT.1/INTER_XFER must be selected in the second selection. FPT_ITT.1/INTER_XFER must be included in the ST for all internal channels used to transfer TSF data between TOE components other than an MDM agent.

Evaluation Activities

FTP_ITC_EXT.1
TSS
The evaluator shall ensure that the TSS contains whether the MDM server communication channel is internal or external to the TOE.

Guidance
TBD

Tests
This testing can be completed in conjunction with the testing for FPT_ITT.1/INTER_XFER / FPT_ITT.1/INTER_XFER_AGENT, FTP_ITC.1/INTER_TSF_XFER_AGENT or FTP_ITC.1(3).

FTP_TRP.1/TRUSTPATH_ENROLL Trusted Path (for Enrollment)

FTP_TRP.1.1/TRUSTPATH_ENROLL

The TSF shall [**selection**:

- ***invoke platform-provided functionality to use [selection:***
  - ***TLS***
  - ***HTTPS***

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

  ]

] **to** provide a **trusted** communication channel between itself and **another trusted IT product** that is logically distinct from other communication **channels** and provides assured identification of its endpoints and protection of the **channel** data from [*modification or disclosure*].

FTP_TRP.1.2/TRUSTPATH_ENROLL

The TSF shall [**selection**: ***invoke platform-provided functionality***, ***implement functionality***] **to** permit [**selection**: *the TSF*, ***another trusted IT product***] to initiate communication via the trusted path.

FTP_TRP.1.3/TRUSTPATH_ENROLL

The TSF shall [**selection**: ***invoke platform-provided functionality*** , ***implement functionality***] **to initiate communication via the trusted channel** for **[assignment: *list of functions for which a trusted channel is required*]** .

Application Note: This requirement ensures that authorized MD users initiate all communication with the TOE via a trusted path, and that all communications with the TOE by MD users is performed over this path. The purpose of this connection is for enrollment by the MD user. The data passed in this trusted communication channel are encrypted as defined in the protocol chosen in the first selection. The ST author chooses the mechanism or mechanisms supported by the TOE.

If the ST author selects "TLS as defined in the Package for Transport Layer Security" the TSF must be validated against requirements from the Package for Transport Layer Security, with the following selections made:

- FCS_TLS_EXT.1:
  - TLS must be selected
  - server must be selected
- FCS_TLSS_EXT.1.1:
  - The cipher suites selected must correspond with the algorithms and hash functions allowed in FCS_COP.1.

Protocol, RBG, Certificate validation, algorithm, and similar services may be met with platform-provided services.

If HTTPS is chosen, FCS_HTTPS_EXT.1 must be included in the ST.

Evaluation Activities

FTP_TRP.1/TRUSTPATH_ENROLL
TSS
The evaluator shall examine the TSS to determine that the methods of remote enrollment are indicated, along with how those communications are protected. The evaluator shall also confirm that all protocols listed in the TSS in support of enrollment are consistent with those specified in the requirement, and are included in the requirements in the ST.

If "invoke platform-provided functionality" is selected, the evaluator shall examine the TSS to verify that it describes (for each supported platform) how this functionality is invoked (it should be noted that this may be through a mechanism that is not implemented by the MDM server; nonetheless, that mechanism will be identified in the TSS as part of this evaluation activity).

Guidance
The evaluator shall confirm that the operational guidance contains instructions for establishing the enrollment sessions for each supported method.

Tests
For each MDM agent or platform listed as supported in the ST:

- *Test FTP_TRP.1/TRUSTPATH_ENROLL:1: The evaluator shall ensure that communications using each specified (in the operational guidance) enrollment method is tested during the course of the evaluation, setting up the connections as described in the operational guidance and ensuring that communication is successful.*
- *Test FTP_TRP.1/TRUSTPATH_ENROLL:2: For each method of enrollment supported, the evaluator shall follow the operational guidance to ensure that there is no available interface that can be used by a remote user to establish enrollment sessions without invoking the trusted path.*
- *Test FTP_TRP.1/TRUSTPATH_ENROLL:3: The evaluator shall ensure, for each method enrollment, the channel data is not sent in plaintext.*

Further evaluation activities are associated with the specific protocols.

FTP_TRP.1/TRUSTPATH_JOIN Trusted Path (for Joining)

This is an objective component.

FTP_TRP.1.1/TRUSTPATH_JOIN

The TSF shall [**selection**: ***invoke platform-provided functionality*** , ***implement functionality***] **to** provide a communication path between itself and **a joining component** that is logically distinct from other communication paths and provides assured identification of [**selection**: ***the TSF endpoint*** , ***both joining component and TSF endpoint***] and protection of the communicated data from [*modification*] **and** [**selection**: *disclosure*, ***none***].

FTP_TRP.1.2/TRUSTPATH_JOIN

The TSF shall [**selection**: ***invoke platform-provided functionality***, ***implement functionality***] **to** permit [**selection**: *the TSF*, ***the joining component***] to initiate communication via the trusted path.

FTP_TRP.1.3/TRUSTPATH_JOIN

The TSF shall [**selection**: ***invoke platform-provided functionality***, ***implement functionality***] **to** require the use of the trusted path for [*joining components to the TSF under environmental constraints identified in [**assignment**: reference to operational guidance]]*.

Application Note: This SFR implements one of the types of channel identified in the main selection for FCO_CPC_EXT.1.2. The "joining component" in FTP_TRP.1/TRUSTPATH_JOIN is the IT entity that is attempting to join the distributed TOE by using the registration process.

The effect of this SFR is to require the ability for components to communicate in a secure manner while the distributed TSF is being created (or when adding components to an existing distributed TSF). When creating the TSF from the initial pair of components, either of these components may be identified as the TSF for the purposes of satisfying the meaning of "TSF" in this SFR.

The selection at the end of FTP_TRP.1/TRUSTPATH_JOIN recognises that in some cases confidentiality (i.e., protection of the data from disclosure) may not be provided by the channel. The ST author distinguishes in the TSS whether in this case the TOE relies on the environment to provide confidentiality (as part of the constraints referenced in FTP_TRP.1.3/TRUSTPATH_JOIN) or whether the registration data exchanged does not require confidentiality (in which case this assertion must be justified). If "none" is selected, then this word may be omitted in the ST to improve readability.

The assignment in FTP_TRP.1.3/TRUSTPATH_JOIN ensures that the ST highlights any specific details needed to protect the registration environment. Note that when the ST uses FTP_TRP.1/TRUSTPATH_JOIN for the registration channel then this channel cannot be reused as the normal inter-component communication channel (the latter channel must meet FTP_ITC.1 or FPT_ITT.1/INTER_XFER / FPT_ITT.1/INTER_XFER_AGENT). Specific requirements for Preparative Procedures relating to FTP_TRP.1/TRUSTPATH_JOIN are defined in the Evaluation Activities.

Evaluation Activities

FTP_TRP.1/TRUSTPATH_JOIN
TSS
The evaluator shall examine the TSS to determine that the methods of joining TOE components are indicated, along with how those communications are protected. The evaluator shall also confirm that all protocols listed in the TSS in support of joining are consistent with those specified in the requirement, and are included in the requirements in the ST.

If "invoke platform-provided functionality" is selected, the evaluator shall examine the TSS to verify that it describes (for each supported platform) how this functionality is invoked (it should be noted that this may be through a mechanism that is not implemented by the MDM server; nonetheless, that mechanism will be identified in the TSS as part of this evaluation activity).

Guidance
The evaluator shall confirm that the operational guidance contains instructions for joining TOE components for each supported method.

Tests
The evaluator shall also perform the following tests:

- *Test FTP_TRP.1/TRUSTPATH_JOIN:1: The evaluator shall ensure that the communications path for joining components to the TSF is tested for each distinct (nonequivalent) component type, setting up the connections as described in the guidance documentation and ensuring that communication is successful. In particular the evaluator shall confirm that requirements on environment protection for the registration process are consistent with observations made on the test configuration (for example, a requirement to isolate the components from the Internet during registration might be inconsistent with the need for a component to contact a license server). If no requirements on the registration environment are identified as necessary to protect confidentiality, then the evaluator shall confirm that the key used for registration can be configured (following the instructions in the guidance documentation) to be at least the same length as the key used for the internal TSF channel that is being enabled. The evaluator shall confirm that the key used for the channel is unique to the pair of components (this is done by identifying the relevant key during the registration test: it is not necessary to examine the key value).*
- *Test FTP_TRP.1/TRUSTPATH_JOIN:2: The evaluator shall follow the guidance documentation to ensure that in fact the communication channel can be enabled by an administrator for all the TOE components identified in the guidance documentation as capable of initiation.*
- *Test FTP_TRP.1/TRUSTPATH_JOIN:3: The evaluator shall ensure that if the guidance documentation states that the channel data is encrypted then the data observed on the channel is not plaintext.*
- *Test FTP_TRP.1/TRUSTPATH_JOIN:4: The evaluator shall ensure that, for each different pair of nonequivalent component types that can use the registration channel, the connection is physically interrupted during a joining attempt. The evaluator shall ensure that when physical connectivity is restored, communications are appropriately protected.*

Further evaluation activities are associated with the specific protocols.

FTP_TRP.1/TRUSTPATH_REM_ADMIN Trusted Path (for Remote Administration)

FTP_TRP.1.1/TRUSTPATH_REM_ADMIN

The TSF shall [**selection**:

- ***invoke platform-provided functionality to use [selection:***
  - ***IPsec***
  - ***TLS***
  - ***HTTPS***
  - ***SSH***

  ]
- ***implement functionality using [selection:***
  - ***IPsec as defined in the PP-Module for VPN Client***
  - ***TLS as defined in the Package for Transport Layer Security***
  - ***HTTPS in accordance with FCS_HTTPS_EXT.1***
  - ***SSH as defined in the Functional Package for Secure Shell***

  ]

] **to** provide a communication path between itself **as a [selection:** ***server, peer*****]** 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/TRUSTPATH_REM_ADMIN

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

FTP_TRP.1.3/TRUSTPATH_REM_ADMIN

The TSF shall require the use of the trusted path for [*initial* ***administrator*** *authentication, [remote administration]*].

Application Note: This requirement ensures that authorized remote administrators initiate all communication with the TOE via a trusted path, and that all communications with the TOE by remote administrators is performed over this path. The data passed in this trusted communication channel are encrypted as defined in the protocol chosen in the first selection. The ST author chooses the mechanism or mechanisms supported by the TOE.

If "IPsec as defined in the PP-Module for VPN Client" is selected, the TSF must claim conformance to a PP-Configuration that includes the VPN Client PP-Module.

If the ST author selects "SSH as defined in the Functional Package for Secure Shell," the TSF must be validated against the FP for Secure Shell with the MDM PP. It should be noted that due to constraints imposed by this PP that SHA-1 cannot be used.

If the ST author selects "TLS as defined in the Package for Transport Layer Security" the TSF must be validated against requirements from the Package for Transport Layer Security, with the following selections made:

- FCS_TLS_EXT.1:
  - TLS must be selected
  - Server must be selected
- FCS_TLSS_EXT.1.1:
  - The cipher suites selected must correspond with the algorithms and hash functions allowed in FCS_COP.1.

Protocol, RBG, Certificate validation, algorithm, and similar services may be met with platform-provided services.

If HTTPS is chosen, FCS_HTTPS_EXT.1 must be included in the ST.

Evaluation Activities

FTP_TRP.1/TRUSTPATH_REM_ADMIN

TSS

The evaluator shall examine the TSS to determine that the methods of remote TOE administration are indicated, along with how those communications are protected. The evaluator shall also confirm that all protocols listed in the TSS in support of TOE administration are consistent with those specified in the requirement, and are included in the requirements in the ST.

If "invoke platform-provided functionality" is selected, the evaluator shall examine the TSS to verify that it describes (for each supported platform) how this functionality is invoked (it should be noted that this may be through a mechanism that is not implemented by the MDM server; nonetheless, that mechanism will be identified in the TSS as part of this evaluation activity).

Guidance

The evaluator shall confirm that the operational guidance contains instructions for establishing the remote administrative sessions for each supported method.

Tests

The evaluator shall also perform the following tests:

- *Test FTP_TRP.1/TRUSTPATH_REM_ADMIN:1: The evaluator shall ensure that communications using each specified (in the operational guidance) remote administration method is tested during the course of the evaluation, setting up the connections as described in the operational guidance and ensuring that communication is successful.*
- *Test FTP_TRP.1/TRUSTPATH_REM_ADMIN:2: For each method of remote administration supported, the evaluator shall follow the operational guidance to ensure that there is no available interface that can be used by a remote user to establish remote administrative sessions without invoking the trusted path.*
- *Test FTP_TRP.1/TRUSTPATH_REM_ADMIN:3: The evaluator shall ensure, for each method of remote administration, the channel data is not sent in plaintext.*

Further evaluation activities are associated with the specific protocols.

4.1.10 TOE Security Functional Requirements Rationale

The following rationale provides justification for each security objective for the TOE, showing that the SFRs are suitable to meet and achieve the security objectives:

Table 5: SFR Rationale

Objective	Addressed by	Rationale

4.2 Security Assurance Requirements

The Security Objectives for the TOE in Section 4 Security Requirements were constructed to address threats identified in Section 2.1 Threats. The Security Functional Requirements (SFRs) in Section 4 Security Requirements are a formal instantiation of the Security Objectives. The PP identifies the Security Assurance Requirements (SARs) to frame the extent to which the evaluator assesses the documentation applicable for the evaluation and performs independent testing.
This section lists the set of SARs from CC part 3 that are required in evaluations against this PP. Individual Evaluation Activities (AAs) to be performed are specified both in Section 4 Security Requirements as well as in this section.

The general model for evaluation of TOEs against STs written to conform to this PP is as follows:

After the ST has been approved for evaluation, the ITSEF will obtain the TOE, supporting environmental IT, and the administrative and user guides for the TOE. The ITSEF is expected to perform actions mandated by the Common Evaluation Methodology (CEM) for the ASE and ALC SARs. The ITSEF also performs the Evaluation Activities contained within Section 4 Security Requirements, which are intended to be an interpretation of the other CEM assurance requirements as they apply to the specific technology instantiated in the TOE. The Evaluation Activities that are captured in Section 4 Security Requirements also provide clarification as to what the developer needs to provide to demonstrate the TOE is compliant with the PP.

The TOE security assurance requirements are identified in .

Assurance Class	Assurance Components
Security Target (ASE)	ST introduction (ASE_INT.1) Conformance claims (ASE_CCL.1) Security objectives for the operational environment (ASE_OBJ.1) Extended components definition (ASE_ECD.1) Stated security requirements (ASE_REQ.1) TOE summary specification (ASE_TSS.1)
Development (ADV)	Basic functional specification (ADV_FSP.1)
Guidance documents (AGD)	Operational user guidance (AGD_OPE.1) Preparative procedures (AGD_PRE.1)
Life cycle support (ALC)	Labeling of the TOE (ALC_CMC.1) TOE CM coverage (ALC_CMS.1)
Tests (ATE)	Independent testing - sample (ATE_IND.1)
Vulnerability assessment (AVA)	Vulnerability survey (AVA_VAN.1)

4.2.1 Class ASE: Security Target

The ST is evaluated as per ASE activities defined in the CEM. In addition, there may be Evaluation Activities specified within Section 4 Security Requirements and relevant appendices that call for necessary descriptions to be included in the TSS that are specific to the TOE technology type.

ASE_TSS.1 TOE summary specification

Developer action elements:

Content and presentation elements:

ASE_TSS.1.1C
The TOE summary specification shall describe how the TOE meets each SFR.

Evaluator action elements:

Evaluation Activities

ASE_TSS.1

4.2.2 Class ADV: Development

The design information about the TOE is contained in the guidance documentation available to the end user as well as the TSS portion of the ST, and any additional information required by this PP that is not to be made public (e.g., Entropy Essay).

ADV_FSP.1 Basic Functional Specification (ADV_FSP.1)

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

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

Developer action elements:

ADV_FSP.1.1D
: The developer shall provide a functional specification.

ADV_FSP.1.2D
: The developer shall provide a tracing from the functional specification to the SFRs.

: **Application Note:** As indicated in the introduction to this section, the functional specification is comprised of the information contained in the AGD_OPE and AGD_PRE documentation. The developer may reference a website accessible to application developers and the evaluator. The evaluation activities in the functional requirements point to evidence that should exist in the documentation and TSS section; since these are directly associated with the SFRs, the tracing in element ADV_FSP.1.2D is implicitly already done and no additional documentation is necessary.

Content and presentation elements:

ADV_FSP.1.1C
: The functional specification shall describe the purpose and method of use for each SFR-enforcing and SFR-supporting TSFI.

ADV_FSP.1.2C
: The functional specification shall identify all parameters associated with each SFR-enforcing and SFR-supporting TSFI.

ADV_FSP.1.3C
: The functional specification shall provide rationale for the implicit categorization of interfaces as SFR-non-interfering.

ADV_FSP.1.4C
: The tracing shall demonstrate that the SFRs trace to TSFIs in the functional specification.

Evaluator action elements:

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

ADV_FSP.1.2E
: The evaluator shall determine that the functional specification is an accurate and complete instantiation of the SFRs.

Evaluation Activities

ADV_FSP.1
There are no specific evaluation activities associated with these SARs, except ensuring the information is provided. The functional specification documentation is provided to support the evaluation activities described in Section 5 and the relevant appendices, and other activities

described for AGD, ATE, and AVA SARs. The requirements on the content of the functional specification information is implicitly assessed by virtue of the other evaluation activities being performed; if the evaluator is unable to perform an activity because there is insufficient interface information, then an adequate functional specification has not been provided.

4.2.3 Class AGD: Guidance Documentation

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

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

- instructions to successfully install the TSF in that environment
- instructions to manage the security of the TSF as a product and as a component of the larger operational environment
- instructions to provide a protected administrative capability

Guidance pertaining to particular security functionality must also be provided; requirements on such guidance are contained in the evaluation activities specified with each requirement.

AGD_OPE.1 Operational User Guidance (AGD_OPE.1)

Developer action elements:

AGD_OPE.1.1D

The developer shall provide operational user guidance.

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

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

Content and presentation elements:

AGD_OPE.1.1C

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

Application Note: User and administrator are to be considered in the definition of user role.

AGD_OPE.1.2C

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

AGD_OPE.1.3C

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

AGD_OPE.1.4C

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

AGD_OPE.1.5C

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

AGD_OPE.1.6C

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

AGD_OPE.1.7C

The operational user guidance shall be clear and reasonable.

Evaluator action elements:

AGD_OPE.1.1E

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

Evaluation Activities

AGD_OPE.1
Some of the contents of the operational guidance will be verified by the evaluation activities in Sections 4.2, 4.3, and 4.4, and evaluation of the TOE according to the CEM. The following additional information is also required.

If cryptographic functions are provided by the TOE, the operational guidance shall contain instructions for configuring the cryptographic engine associated with the evaluated configuration of the TOE. It shall provide a warning to the administrator that use of other cryptographic engines was not evaluated nor tested during the CC evaluation of the TOE.

The documentation must describe the process for verifying updates to the TOE by verifying a digital signature - this may be done by the TOE or the underlying platform. The evaluator shall verify that this process includes the following steps:

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

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

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

4.2.4 Preparative Procedures (AGD_PRE.1)

AGD_PRE.1 Preparative Procedures (AGD_PRE.1)

Developer action elements:

AGD_PRE.1.1D

The developer shall provide the TOE, including its preparative procedures.

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

Content and presentation elements:

AGD_PRE.1.1C

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

AGD_PRE.1.2C

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

Application Note: It is recognised that the application of these requirements will vary depending on aspects such as whether the TOE is delivered in an operational state, or whether it has to be installed at the TOE owner's site, etc.

It might also be the case that no installation is necessary, for example as a Software as a Service implementation in a Cloud environment. In this case it may be inappropriate to require and analyse installation procedures and thus this requirement is implicitly satisfied.

Evaluator action elements:

AGD_PRE.1.1E

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

AGD_PRE.1.2E

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

Evaluation Activities

AGD_PRE.1
As indicated in the introduction above, there are significant expectations with respect to the documentation, especially when configuring the operational environment to support TOE functional requirements. The evaluator shall check to ensure that the guidance provided for the TOE adequately addresses all platforms claimed for the TOE in the ST.

4.2.5 Class ALC: Life-cycle Support

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

ALC_CMC.1 Labeling of the TOE (ALC_CMC.1)

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

Developer action elements:

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

Content and presentation elements:

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

Evaluator action elements:

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

Evaluation Activities

ALC_CMC.1
The evaluator shall check the ST to ensure that it contains an identifier (such as a product name and version number) that specifically identifies the version that meets the requirements of the ST. Further, the evaluator shall check the operational guidance and TOE samples received for testing to ensure that the version number is consistent with that in the ST. If the vendor maintains a website advertising the TOE, the evaluator shall examine the information on the website to ensure that the information in the ST is sufficient to distinguish the product.

4.2.6 TOE CM Coverage (ALC_CMS.1)

ALC_CMS.1 TOE CM Coverage (ALC_CMS.1)

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

Developer action elements:

ALC_CMS.1.1D
The developer shall provide a configuration list for the TOE.

Content and presentation elements:

ALC_CMS.1.1C
The configuration list shall include the following: the TOE itself; and the evaluation evidence required by the SARs.

ALC_CMS.1.2C
The configuration list shall uniquely identify the configuration items.

Evaluator action elements:

ALC_CMS.1.1E

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

Application Note: In cases where the MDM software is Software as a Service, running in a cloud environment where they have little to no control of the operating system and underlying hardware, the evaluated configuration is considered a snapshot of the MDM software with the OS or VM versions used at the time of testing.

Evaluation Activities

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

Life-cycle support is targeted aspects of the developer's life-cycle and instructions to providers of applications for the developer's devices, rather than an in-depth examination of the TSF manufacturer's development and configuration management process. This is not meant to diminish the critical role that a developer's practices play in contributing to the overall trustworthiness of a product; rather, it's a reflection on the information to be made available for evaluation.

The evaluator shall ensure that the TSF is uniquely identified (with respect to other products from the TSF vendor), and that documentation provided by the developer in association with the requirements in the ST is associated with the TSF using this unique identification.

4.2.7 Class ATE: Tests

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

Since many of the APIs are not exposed at the user interface (e.g., touch screen), the ability to stimulate the necessary interfaces requires a developer's test environment. This test environment will allow the evaluator, for example, to access APIs and view file system information that is not available on consumer devices.

ATE_IND.1 Independent Testing - Conformance (ATE_IND.1)

Testing is performed to confirm the functionality described in the TSS as well as the administrative (including configuration and operational) documentation provided. The focus of the testing is to confirm that the requirements specified in Sections 5.3 and 5.4 are being met, although some additional testing is specified for SARs in Section 6. The Evaluation Activities identify the additional testing activities associated with these components. The evaluator produces a test report documenting the plan for and results of testing, as well as coverage arguments focused on the TOE or platform combinations that are claiming conformance to this PP.

Developer action elements:

ATE_IND.1.1D

The developer shall provide the TOE for testing.

Content and presentation elements:

ATE_IND.1.1C

The TOE shall be suitable for testing.

Evaluator action elements:

ATE_IND.1.1E

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

ATE_IND.1.2E

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

Evaluation Activities

ATE_IND.1
The evaluator shall prepare a test plan and report documenting the testing aspects of the system. The test plan covers all of the testing actions contained in the CEM and the body of this PP's Evaluation Activities. While it is not necessary to have one test case per test listed in an evaluation activity, the evaluator must document in the test plan that each applicable testing requirement in the ST is covered.

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

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

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

4.2.8 Class AVA: Vulnerability Analysis

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

AVA_VAN.1 Vulnerability Survey (AVA_VAN.1)

Developer action elements:

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

Content and presentation elements:

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

Evaluator action elements:

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

AVA_VAN.1.2E
The evaluator shall perform a search of public domain sources to identify potential vulnerabilities in the TOE.

AVA_VAN.1.3E
The evaluator shall conduct penetration testing, based on the identified potential vulnerabilities, to determine that the TOE is resistant to attacks performed by an attacker possessing Basic attack potential.

Evaluation Activities

AVA_VAN.1
As with ATE_IND, the evaluator shall generate a report to document their findings with respect to this requirement. This report could physically be part of the overall test report mentioned in

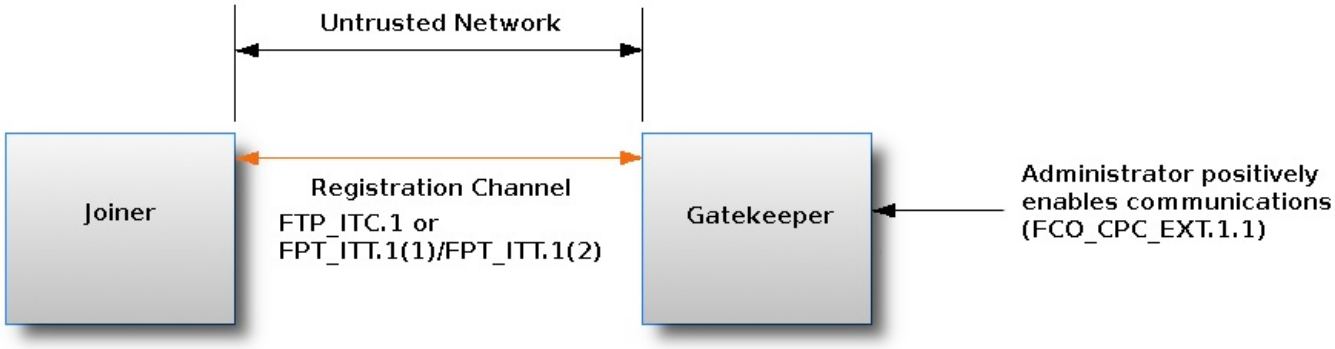
ATE_IND, or a separate document. The evaluator performs a search of public information to determine the vulnerabilities that have been found in network infrastructure devices and the implemented communication protocols in general, as well as those that pertain to the particular TOE. The evaluator documents the sources consulted and the vulnerabilities found in the report. For each vulnerability found, the evaluator either provides a rationale with respect to its non-applicability, or the evaluator formulates a test (using the guidelines provided in ATE_IND) to confirm the vulnerability, if suitable. Suitability is determined by assessing the attack vector needed to take advantage of the vulnerability. If exploiting the vulnerability requires expert skills and an electron microscope, for instance, then a test would not be suitable and an appropriate justification would be formulated.

5 Introduction to Distributed TOEs

This PP includes support for distributed MDM TOEs. MDMs can sometimes be composed of multiple components operating as a logical whole. Frequently we see this architecture when dealing with products hosted in the cloud and offered as Software as a Service. There are a number of different architectures; but fundamentally, they are variations of the following model where the SFRs of this PP can only be fulfilled if the components are deployed and operate together. To be considered a distributed TOE, a minimum of two interconnected components are required.

5.1 Registration of Distributed TOE Components

When dealing with a distributed TOE, a number of separate components need to be brought together in the operational environment in order to create the TOE. This requires that trusted communications channels are set up between certain pairs of components (it is assumed that all components need to communicate with at least one other component, but not that all components need to communicate with all other components). The underlying model for creation of the TOE is to have a "registration process" in which components "join" the TOE. The registration process starts with two components, one of which (the "joiner") is about to join an existing TOE by registering with the other (the "gatekeeper"). The two components will use one or more specified authentication and communication channel options so that the components authenticate each other and protect any sensitive data that is transmitted during the registration process (e.g., a key might be sent by a gatekeeper to the joiner as a result of the registration). The following figures illustrate the three supported registration models. Figure 2 illustrates a distributed TOE registration approach which uses an instance of FPT_ITT.1/INTER_XFER / FPT_ITT.1/INTER_XFER_AGENT or FTP_ITC.1/INTER_XFER_IT to protect the registration exchange.



1) Registration may be performed over any untrusted network
2) Registration performed over IPsec, TLS, SSH, or HTTPS channel
3) Choose FPT_ITT.1(1) if certificate revocation checking is not performed
4) Choose FPT_ITT.1(2) if registration is between the TSF and an MDM agent that is included in the TOE
5) Choose FTP_ITC.1 if certificate revocation checking is performed
6) Registration channel may be re-used for internal TSF communications

Figure 2: Distributed TOE registration using channel satisfying FPT_ITT.1/INTER_XFER / FPT_ITT.1/INTER_XFER_AGENT or FTP_ITC.1/INTER_XFER_IT

The second approach (Figure 3) uses an alternative registration channel and supports use cases where the channel relies on environmental security constraints to provide the necessary protection of the registration exchange.

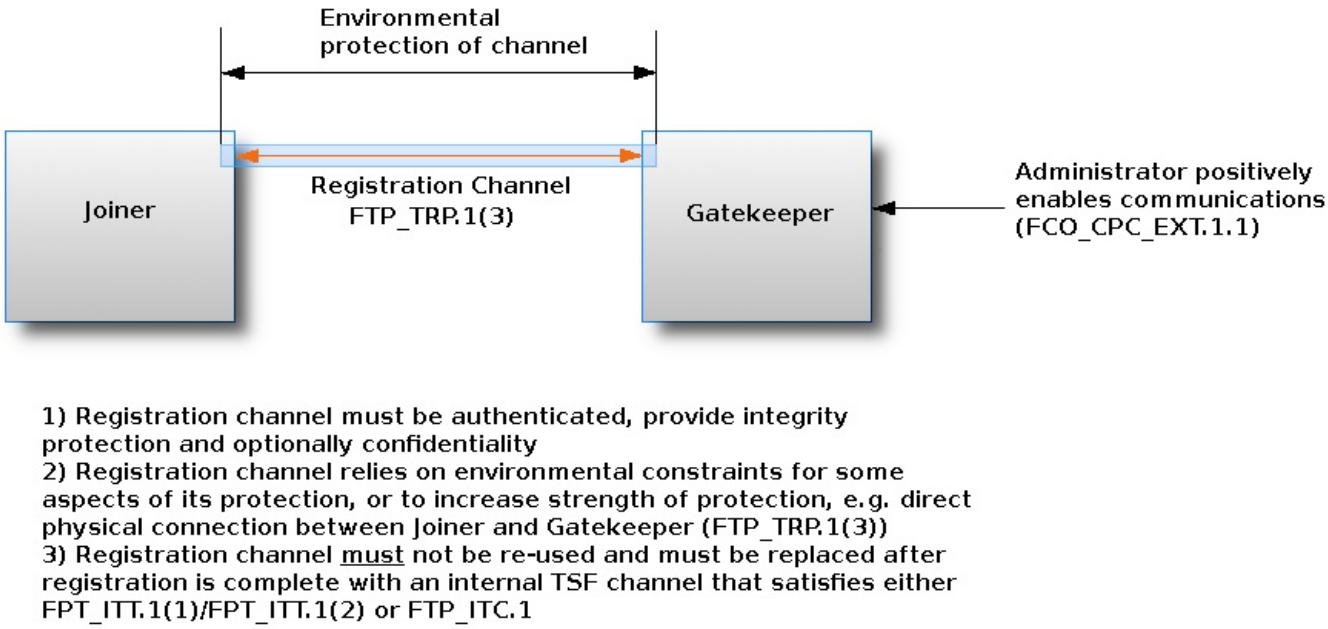


Figure 3: Distributed TOE registration using channel satisfying FTP_TRP.1/TRUSTPATH_JOIN

The final approach (Figure 4) supports use cases where registration is performed manually through direct configuration of both the Joiner and Gatekeeper devices. Once configured, the two components establish an internal TSF channel that satisfies FPT_ITT.1/INTER_XFER / FPT_ITT.1/INTER_XFER_AGENT or FTP_ITC.1/INTER_XFER_IT.

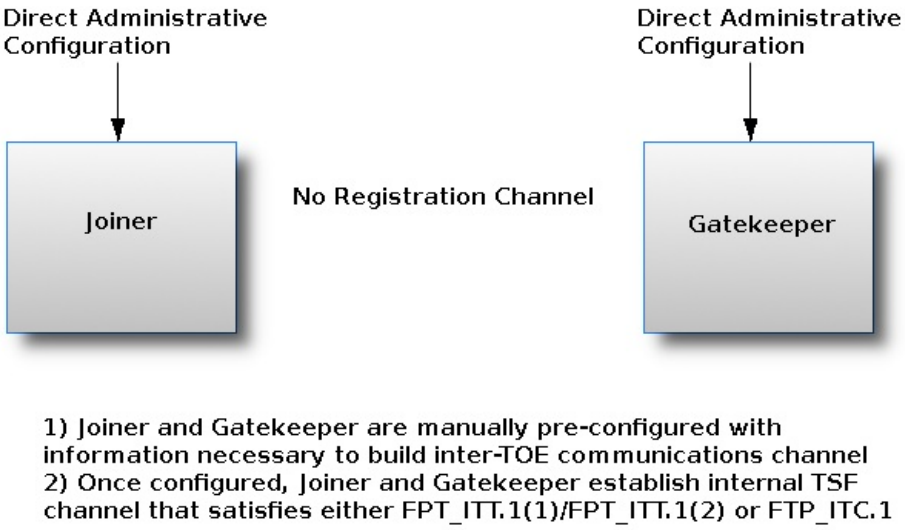


Figure 4: Distributed TOE registration without a registration channel

In each case, during the registration process, the administrator must positively enable the joining components before it can act as part of the TSF. Figure 5 illustrates the approaches that this enablement step may take;

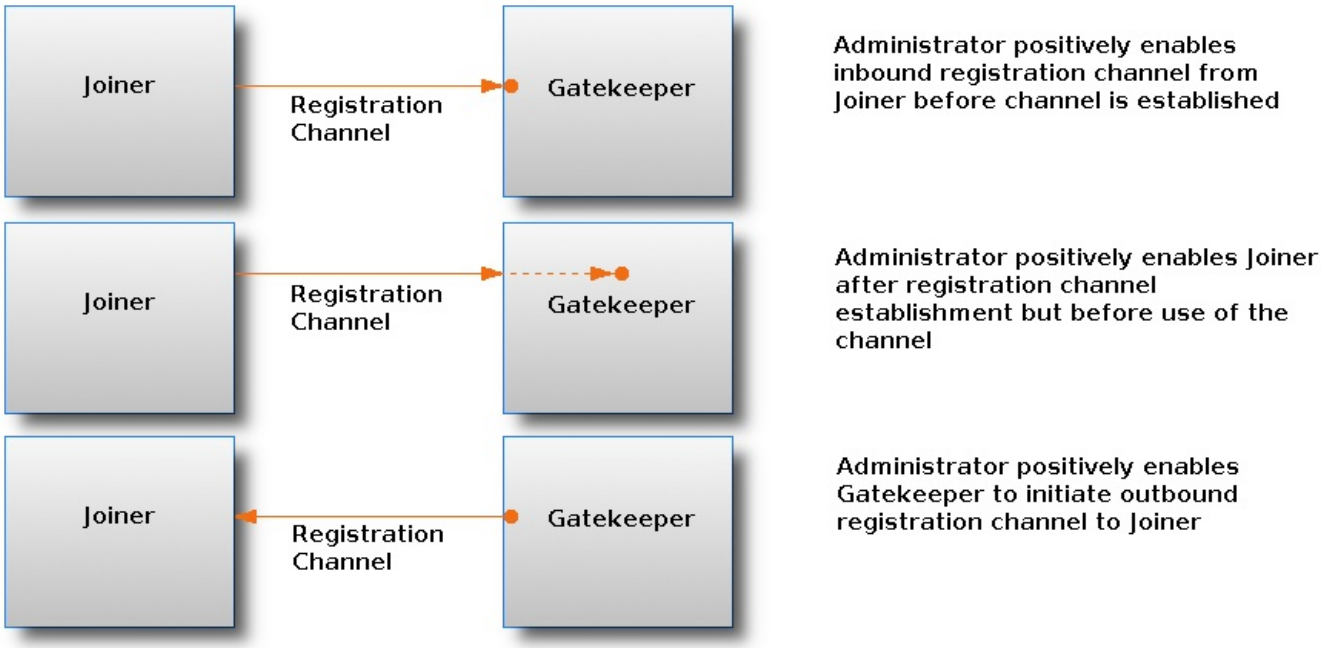


Figure 5: Joiner enablement options for Distributed TOEs

Note that in the case where no registration channel is required, that is the joiner and gatekeeper are directly configured (Figure 4), enablement is implied as part of this direct configuration process.

After registration, the components will communicate between themselves using a normal SSH, TLS, DTLS, IPsec, or HTTPS channel (which is specified in an ST as an instance of FPT_ITT.1/INTER_XFER / FPT_ITT.1/INTER_XFER_AGENT or FTP_ITC.1/INTER_XFER_IT in terms of Section 4 Security Requirements and Table t-audit-optional). This channel for inter-component communications is specified at the top level with the new (extended) SFR FCO_CPC_EXT.1 and is in addition to the other communication channels required for communication with entities outside the TOE (which are specified in an ST as instances of FTP_ITC.1/INTER_XFER_IT and FTP_TRP.1/TRUSTPATH_REM_ADMIN.

5.2 Allocation of Requirements in Distributed TOEs

For a distributed TOE, the security functional requirements in this PP need to be met by the TOE as a whole, but not all SFRs will necessarily be implemented by all components. The following categories are defined in order to specify when each SFR must be implemented by a component:

- **All Components ("All")** - All components that comprise 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 (note that the requirement to meet the PP as a whole requires that at least one component implements these requirements if they are specified in Section 4 Security Requirements).

Table 6 specifies how each of the SFRs in this PP must be met, using the categories above.

Table 6: Security Functional Requirements for Distributed TOEs

Requirement	Description	Distributed TOE SFR Allocation
FAU_ALT_EXT.1	Server Alerts	One
FAU_CRP_EXT.1	Support for Compliance Reporting of Mobile Device Configuration	One
FAU_GEN.1/AUDITGEN	Audit Data Generation	All
FAU_GEN.1/MAS_SERVER	Audit Data Generation	Feature Dependent
FAU_NET_EXT.1	Network Reachability Review	One
FAU_SAR.1	Audit Review	Feature Dependent
FAU_SEL.1	Security Audit Event Selection	One
FAU_STG.1	External Trail Storage	All

FAU_STG.2	Audit Event Storage	Feature Dependent
FCO_CPC_EXT.1	Communication Partner Control	All
FCS_CKM.1	Cryptographic Key Generation	Feature Dependent
FCS_CKM.2	Cryptographic Key Establishment	All
FCS_CKM.6	Cryptographic Key Destruction	All
FCS_COP.1.1/CONF_ALG	Cryptographic Operation (Confidentiality Algorithms)	All
FCS_COP.1.1/HASH_ALG	Cryptographic Operation (Hashing Algorithms)	All
FCS_COP.1.1/KEY_HASH	Cryptographic Operation (Keyed-Hash Message Authentication)	All
FCS_COP.1.1/SIGN_ALG	Cryptographic Operation (Signature Algorithms)	All
FCS_HTTPS_EXT.1	HTTPS Protocol	Feature Dependent
FCS_IV_EXT.1	Initialization Vector Generation	Feature Dependent
FCS_RBG.1	Random Bit Generation (RBG)	All
FCS_RBG.2	Random Bit Generation (External Seeding)	Feature Dependent
FCS_RBG.3	Random Bit Generation (Internal Seeding - Single Source)	Feature Dependent
FCS_RBG.4	Random Bit Generation (Internal Seeding - Multiple Sources)	Feature Dependent
FCS_RBG.5	Random Bit Generation (Combining Noise Sources)	Feature Dependent
FCS_STG_EXT.1	Cryptographic Key Storage	All
FCS_STG_EXT.2	Encrypted Cryptographic Key Storage	Feature Dependent
FIA_CLI_EXT.1	Client Authorization	One
FIA_ENR_EXT.1	Enrollment of Mobile Device into Management	One
FIA_TOK_EXT.1	Client Tokens	One
FIA_UAU.1	Timing of Authentication	One
FIA_UAU.4	Single-Use Authentication Mechanisms	One
FIA_X509_EXT.1/CERTVAL_MAN	X.509 Certification Validation	Feature Dependent
FIA_X509_EXT.1/CERTVAL_SEL	X.509 Certification Validation	Feature Dependent
FIA_X509_EXT.2	X.509 Certificate Authentication	Feature Dependent
FIA_X509_EXT.3	X.509 Enrollment	Feature Dependent
FIA_X509_EXT.4	Alternate X.509 Enrollment	Feature Dependent
FMT_MOF.1/FUNCBE	Management of functions behaviour	Feature Dependent
FMT_MOF.1/MANAGEMENT_ENROLL	Management of functions behaviour (Enrollment)	Feature Dependent
FMT_MOF.1/MANAGEMENT_MAS	Management of Functions in (MAS Server Downloads)	Feature Dependent
FMT_POL_EXT.1	Trusted Policy Update	One
FMT_SAE_EXT.1	Security Attribute Expiration	One
FMT_SMF.1/MAS	Specification of Management Functions (MAS Server)	Feature Dependent
FMT_SMF.1/SERVER_CONF_AGENT	Specification of Management Functions	One

	(Server configuration of Agent)	
FMT_SMF.1/SERVER_CONF_SERVER	Specification of Management Functions (Server configuration of Server)	Feature Dependent
FMT_SMR.1/SECMAN_ROLES	Security Management Roles	One
FMT_SMR.1/SECMAN_ROLES_MAS	Security Management Roles	Feature Dependent
FPT_API_EXT.1	Use of Supported Services and APIs	All
FPT_FLS.1	Failure with Preservation of Secure State	All
FPT_ITT.1/INTER_XFER	Internal TOE TSF Data Transfer	Feature Dependent
FPT_ITT.1/INTER_XFER_AGENT	Internal TOE TSF Data Transfer (MDM Agent)	Feature Dependent
FPT_LIB_EXT.1	Use of Third-Party Libraries	All
FPT_TST.1	TSF Self-Testing	All
FPT_TST_EXT.1	Functionality Testing	All (except for agent components)
FPT_TUD_EXT.1	Trusted Update	All
FTA_TAB.1	Default TOE Access Banners	One
FTP_ITC.1/INTER_TSF_XFER_AGENT	Inter-TSF Trusted Channel (MDM Agent)	One
FTP_ITC.1/INTER_XFER_IT	Inter-TSF Trusted Channel (Authorized IT Entities)	One
FTP_ITC_EXT.1	Trusted Channel	One
FTP_TRP.1/TRUSTPATH_ENROLL	Trusted Path for Enrollment	Feature Dependent
FTP_TRP.1/TRUSTPATH_JOIN	Trusted Path for Joining	Feature Dependent
FTP_TRP.1/TRUSTPATH_REM_ADMIN	Trusted Path for Remote Administration	Feature Dependent

Only those SFRs included in the ST are required to be audited. The ST for a distributed TOE must include a mapping of SFRs to each of the components of the TOE. (Note that this deliverable is examined as part of the ASE_TSS.1 and AVA_VAN.1 Evaluation Activities.) The ST for a distributed TOE may also introduce a "minimum configuration" and identify components that may have instances added to an operational configuration without affecting the validity of the CC certification. Appendix E describes Evaluation Activities relating to these equivalency aspects of a distributed TOE (and hence what is expected in the ST).

5.3 Security Audit for Distributed TOEs

For distributed TOEs, the handling of audit information might be more complicated than for TOEs consisting only of one component. There are a few basic requirements to be fulfilled:

- Every component must be able to generate audit information.
- Every component must be able to buffer audit information and forward it to another TOE component or an external audit server. Optionally, each component may store audit information locally.
- For the overall TOE it must be possible to send out audit information to an external audit server.

In general, every component must be able to generate its own audit information. It would be possible that every component also stores its own audit information locally as well as every component could be able to send out audit data to an external audit server. It would also be sufficient that every component would be able to generate its own audit data and buffer it locally before the information is sent out to one or more other TOE components for local storage or transmission to an external audit server. For the transfer of audit records between TOE components the secure connection via FTP_ITC.1/INTER_XFER_IT or FPT_ITT.1/INTER_XFER / FPT_ITT.1/INTER_XFER_AGENT must be used.

Such a solution would still be suitable to fulfill the requirement that all audit-related SFRs have to be fulfilled by all TOE components, although formally not every component would support local storage or transfer to an external audit server itself.

Regarding the establishment of inter-TOE communication, error conditions as well as successful connection and tear-down events should be captured by both ends of the connection.

All TOE components shall be able to generate its own audit data according to FAU_GEN.1 for all SFRs that it implements. For distributed TOEs, a mapping shall be provided to show which auditable events according to FAU_GEN.1 are covered by which components (also giving a justification that the records generated by each component cover all the SFRs that it implements). The overall TOE has to provide audit information about all events defined for FAU_GEN.1. As a result, at least one TOE component has to be assigned to every auditable event defined for FAU_GEN.1. The part of the mapping related to Table 2 shall be consistent with the mapping of SFRs to TOE components for ASE_TSS.1 in the sense that all components defined as generating audit information for a particular SFR should also contribute to that SFR in the mapping for ASE_TSS.1. This applies not only to audit events defined for mandatory SFRs but also to all audit events for optional, selection-based, and objective SFRs as defined in Table t-audit-optional, Table t-audit-objective, and Table t-audit-sel-based.

If one or more of the optional audit components FAU_STG.1 or FAU_STG.2 are selected in the ST derived from this PP, then the SFR mapping for ASE_TSS.1 must include a specific identification of the TOE components to which they apply.

Appendix A - Implementation-dependent Requirements

Implementation-dependent Requirements Appendix defines requirements that must be claimed in the ST if the TOE implements particular product features. For this technology type, the following product features require the claiming of additional SFRs:

Appendix B - Extended Component Definitions

This appendix contains the definitions for all extended requirements specified in the PP.

B.1 Extended Components Table

All extended components specified in the PP are listed in this table:

Table 7: Extended Component Definitions

Functional Class	Functional Components
Communication (FCO)	FCO_CPC_EXT Component Registration Channel Definition
Cryptographic Support (FCS)	FCS_HTTPS_EXT HTTPS Protocol FCS_IV_EXT Initialization Vector Generation FCS_STG_EXT Encrypted Cryptographic Key Storage
Identification and Authentication (FIA)	FIA_CLI_EXT Client Authorization FIA_ENR_EXT Enrollment of Mobile Device into Management FIA_TOK_EXT Client Tokens
Protection of the TSF (FPT)	FPT_API_EXT Use of Supported Services and APIs FPT_LIB_EXT Use of Third-Party Libraries FPT_TST_EXT Functionality Testing FPT_TUD_EXT Trusted Update
Security Audit (FAU)	FAU_ALT_EXT Server Alerts FAU_CRP_EXT Support for Compliance Reporting of Mobile Device Configuration FAU_NET_EXT Network Reachability Review
Security Management (FMT)	FMT_POL_EXT Trusted Policy Update FMT_SAE_EXT Security Attribute Expiration
Trusted Path/Channels (FTP)	FTP_ITC_EXT Trusted Channel

B.2 Extended Component Definitions

B.2.1 Communication (FCO)

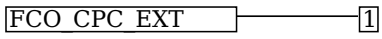
This PP defines the following extended components as part of the FCO class originally defined by CC Part 2:

B.2.1.1 FCO_CPC_EXT Component Registration Channel Definition

Family Behavior

This family describes the registration process, including the capability for the administrator to enable or disable communications between a distributed TOE and other components of the TOE.

Component Leveling



FCO_CPC_EXT.1, Component Registration Channel Definition, defines requirements for the registration process for distributed TOEs.

Management: FCO_CPC_EXT.1

There are no management activities foreseen.

Audit: FCO_CPC_EXT.1

The following actions should be auditable if FAU_GEN security audit data generation is included in the PP or ST.

- Enabling or disabling communications between a pair of components.
- Identities of the endpoint's pairs enabled or disabled.

FCO_CPC_EXT.1 Component Registration Channel Definition

Hierarchical to: No other components.

Dependencies to: FPT_ITT.1 TSF Data Transfer
FTP_TRP.1 Trusted Path

FCO_CPC_EXT.1.1

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

- *A channel that meets the secure channel requirements in [**selection**: FTP_ITC.1, FPT_ITT.1/INTER_XFER, FPT_ITT.1/INTER_XFER_AGENT]*
- *A channel that meets the secure registration channel requirements in [**selection**: FTP_TRP.1/TRUSTPATH_ENROLL, FTP_TRP.1/TRUSTPATH_JOIN]*
- *No channel*

] for at least TSF data.

FCO_CPC_EXT.1.3

The TSF shall [**selection**: *invoke platform-provided functionality*, *implement functionality*] to enable an administrator to disable communications between any pair of TOE components.

B.2.2 Cryptographic Support (FCS)

This PP defines the following extended components as part of the FCS class originally defined by CC Part 2:

B.2.2.1 FCS_HTTPS_EXT HTTPS Protocol

Family Behavior

This family defines requirements for protecting HTTP communications between the TOE and an external IT entity.

Component Leveling

FCS_HTTPS_EXT ——— 1

FCS_HTTPS_EXT.1, HTTPS Protocol, defines requirements for the implementation of the HTTPS protocol.

Management: FCS_HTTPS_EXT.1

There are no management activities foreseen.

Audit: FCS_HTTPS_EXT.1

The following actions should be auditable if FAU_GEN security audit data generation is included in the PP or ST.

- Failure of the certificate validity check.
- Issuer Name and Subject Name of certificate.
- [**selection**: *User's authorization decision*, *No additional information*]

FCS_HTTPS_EXT.1 HTTPS Protocol

Hierarchical to: No other components.

Dependencies to: FCS_TLS_EXT.1 TLS Protocol
[FCS_TLSC_EXT.1 TLS Client Protocol or
FCS_TLSS_EXT.1 TLS Server Protocol

FCS_HTTPS_EXT.1.1

The TSF shall implement the HTTPS protocol that complies with RFC 2818.

FCS_HTTPS_EXT.1.2

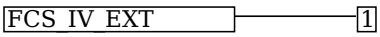
The TSF shall implement HTTPS using TLS in accordance with the Package for Transport Layer Security.

B.2.2.2 FCS_IV_EXT Initialization Vector Generation

Family Behavior

This family defines requirements for generating IVs in accordance with NIST-approved cipher modes.

Component Leveling



FCS_IV_EXT.1, Initialization Vector Generation, defines requirements for generating IVs.

Management: FCS_IV_EXT.1

There are no management activities foreseen.

Audit: FCS_IV_EXT.1

There are no auditable events foreseen.

FCS_IV_EXT.1 Initialization Vector Generation

Hierarchical to: No other components.

Dependencies to: No dependencies.

FCS_IV_EXT.1.1

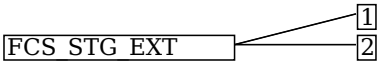
The TSF shall [**selection**: *invoke platform-provided functionality, implement functionality*] to generate IVs in accordance with Table 4.

B.2.2.3 FCS_STG_EXT Encrypted Cryptographic Key Storage

Family Behavior

This family defines requirements for ensuring the protection of keys and secrets.

Component Leveling



FCS_STG_EXT.1, Cryptographic Key Storage, defines requirements for the security of persistent secrets and private keys.

FCS_STG_EXT.2, Encrypted Cryptographic Key Storage, defines requirements for preventing access to private keys and persistent secrets.

Management: FCS_STG_EXT.1

The following actions could be considered for the management functions in FMT.

- Import keys or secrets into the secure key storage (MDF Function 9)

Audit: FCS_STG_EXT.1

There are no auditable events foreseen.

FCS_STG_EXT.1 Cryptographic Key Storage

Hierarchical to: No other components.

Dependencies to: No dependencies.

FCS_STG_EXT.1.1

The TSF shall use [**selection**: *platform-provided key storage, encryption as specified in FCS_STG_EXT.2*] for all persistent secrets and private keys.

Management: FCS_STG_EXT.2

There are no management activities foreseen.

Audit: FCS_STG_EXT.2

There are no auditable events foreseen.

FCS_STG_EXT.2 Encrypted Cryptographic Key Storage

Hierarchical to: No other components.

Dependencies to: No dependencies.

FCS_STG_EXT.2.1

The TSF shall [**selection**: *invoke platform-provided functionality*, *implement functionality*] to encrypt all keys using AES in the [**selection**: *Key Wrap (KW) mode*, *Key Wrap with Padding (KWP) mode*, *GCM*, *CCM*, *CBC mode*].

B.2.3 Identification and Authentication (FIA)

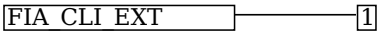
This PP defines the following extended components as part of the FIA class originally defined by CC Part 2:

B.2.3.1 FIA_CLI_EXT Client Authorization

Family Behavior

This family defines requirements for unique certificate or token use.

Component Leveling



FIA_CLI_EXT.1, Client Authorization, defines requirements for a unique certificate or token for each client device.

Management: FIA_CLI_EXT.1

There are no management activities foreseen.

Audit: FIA_CLI_EXT.1

There are no auditable events foreseen.

FIA_CLI_EXT.1 Client Authorization

Hierarchical to: No other components.

Dependencies to: No dependencies.

FIA_CLI_EXT.1.1

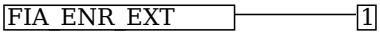
The TSF shall require a unique [**selection**: *certificate*, *token as defined in FIA_TOK_EXT.1*] for each client device.

B.2.3.2 FIA_ENR_EXT Enrollment of Mobile Device into Management

Family Behavior

This family defines requirements for authenticating remote users and limiting user enrollment.

Component Leveling



FIA_ENR_EXT.1, Enrollment of Mobile Device into Management, defines requirements for authenticating and limiting user actions.

Management: FIA_ENR_EXT.1

The following actions could be considered for the management functions in FMT.

- Configure the specific device models.
- Configure the specific time period.

Audit: FIA_ENR_EXT.1

The following actions should be auditable if FAU_GEN security audit data generation is included in the PP or ST.

- Failure of MD user authentication.
- Presented username.

FIA_ENR_EXT.1 Enrollment of Mobile Device into Management

Hierarchical to: No other components.

Dependencies to: FIA_UAU.4 Single-Use Authentication Mechanisms

FIA_ENR_EXT.1.1

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

FIA_ENR_EXT.1.2

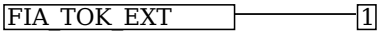
The TSF shall limit the user's enrollment of devices to devices specified by [**selection**: *IMEI*, *[**assignment**: a unique device ID]*] and [**selection**: *specific device models*, *a number of devices*, *specific time period*, *[**assignment**: other features]*, *no other features*].

B.2.3.3 FIA_TOK_EXT Client Tokens

Family Behavior

This family defines requirements for using a unique device to generate unique tokens for client devices.

Component Leveling



FIA_TOK_EXT.1, Client Tokens, defines requirements for generating unique tokens.

Management: FIA_TOK_EXT.1

There are no management activities foreseen.

Audit: FIA_TOK_EXT.1

There are no auditable events foreseen.

FIA_TOK_EXT.1 Client Tokens

Hierarchical to: No other components.

Dependencies to: No dependencies.

FIA_TOK_EXT.1.1

The TSF shall [**selection**: *invoke platform-provided functionality*, *implement functionality*] to use [**selection**: *IMEI*, *[**assignment**: other unique device ID]*] to generate a unique token for each client device.

B.2.4 Protection of the TSF (FPT)

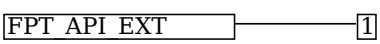
This PP defines the following extended components as part of the FPT class originally defined by CC Part 2:

B.2.4.1 FPT_API_EXT Use of Supported Services and APIs

Family Behavior

This family describes document platform APIs when selecting "invoke platform-provided functionality."

Component Leveling



FPT_API_EXT.1, Use of Supported Services and APIs, defines requirements for API usage.

Management: FPT_API_EXT.1

There are no management activities foreseen.

Audit: FPT_API_EXT.1

There are no auditable events foreseen.

FPT_API_EXT.1 Use of Supported Services and APIs

Hierarchical to: No other components.

Dependencies to: No dependencies.

FPT_API_EXT.1.1

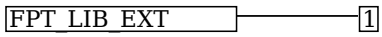
The TSF shall use only documented platform APIs.

B.2.4.2 FPT_LIB_EXT Use of Third-Party Libraries

Family Behavior

This family describes packaging third-party libraries when selecting "implement functionality."

Component Leveling



FPT_LIB_EXT.1, Use of Third-Party Libraries, defines requirements for third-party libraries.

Management: FPT_LIB_EXT.1

There are no management activities foreseen.

Audit: FPT_LIB_EXT.1

There are no auditable events foreseen.

FPT_LIB_EXT.1 Use of Third-Party Libraries

Hierarchical to: No other components.

Dependencies to: No dependencies.

FPT_LIB_EXT.1.1

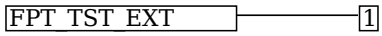
The MDM software shall be packaged with only [**assignment**: *list of third-party libraries*].

B.2.4.3 FPT_TST_EXT Functionality Testing

Family Behavior

This family defines requirements for running self-tests and verifying integrity or executable code.

Component Leveling



FPT_TST_EXT.1, Functionality Testing, defines requirements for the integrity of self-testing.

Management: FPT_TST_EXT.1

There are no management activities foreseen.

Audit: FPT_TST_EXT.1

The following actions should be auditable if FAU_GEN security audit data generation is included in the PP or ST.

- Initiation of self-test.
- Failure of self-test.
- Detected integrity violation

FPT_TST_EXT.1 Functionality Testing

Hierarchical to: No other components.

Dependencies to: FPT_TST.1 TSF Self-Testing

FPT_TST_EXT.1.1

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

FPT_TST_EXT.1.2

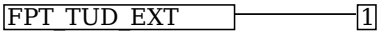
The TSF shall [**selection**: *invoke platform-provided functionality, implement functionality*] to provide the capability to verify the integrity of stored TSF executable code when it is loaded for execution through the use of the [**selection**: *TSF, TOE platform*] -provided cryptographic services.

B.2.4.4 FPT_TUD_EXT Trusted Update

Family Behavior

This family defines requirements for allowing authorized administrators to query software versions, initiate updates, and verify software updates prior to installation.

Component Leveling



FPT_TUD_EXT.1, Trusted Update, defines requirements for authorized administrators to manage software versions and updates.

Management: FPT_TUD_EXT.1

The following actions could be considered for the management functions in FMT.

- Query the current version of the MD firmware or software.
- Update system software (MDF Function 15).

Audit: FPT_TUD_EXT.1

The following action should be auditable if FAU_GEN security audit data generation is included in the PP or ST.

- Success or failure of signature verification

FPT_TUD_EXT.1 Trusted Update

Hierarchical to: No other components.

Dependencies to: No dependencies.

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 [**selection**: *invoke platform-provided functionality*, *implement functionality*] to provide authorized administrators the ability to initiate updates to TSF software.

FPT_TUD_EXT.1.3

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

B.2.5 Security Audit (FAU)

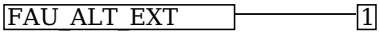
This PP defines the following extended components as part of the FAU class originally defined by CC Part 2:

B.2.5.1 FAU_ALT_EXT Server Alerts

Family Behavior

This family defines requirements for the TSF to alert administrators when a set of specified events occurs.

Component Leveling



FAU_ALT_EXT.1, Server Alerts, defines requirements for alerting the administrator to events.

Management: FAU_ALT_EXT.1

The following actions could be considered for the management functions in FMT.

- Install policies.

Audit: FAU_ALT_EXT.1

The following actions should be auditable if FAU_GEN security audit data generation is include in the PP or ST.

- Type of alert.
- Identity of MD that sent alert.

FAU_ALT_EXT.1 Server Alerts

Hierarchical to: No other components.

Dependencies to: No dependencies.

FAU_ALT_EXT.1.1

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

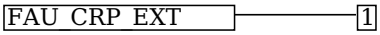
- Change in enrollment status
- Failure to apply policies to a MD
- [**selection**: ***[assignment**: other events]** , *no other events*]

B.2.5.2 FAU_CRP_EXT Support for Compliance Reporting of Mobile Device Configuration

Family Behavior

This family defines requirements for the TSF to provide an interface for the MDM server to convey information about MDs for other systems.

Component Leveling



FAU_CRP_EXT.1, Support for Compliance Reporting of Mobile Device Configuration, defines requirements for providing information to enrolled MDs through a secure channel.

Management: FAU_CRP_EXT.1

The following actions could be considered for the management functions in FMT.

- Query the current version of the MD firmware or software.
- Query the current version of the hardware model of the device.
- Query the current version of installed applications.

Audit: FAU_CRP_EXT.1

There are no auditable events foreseen.

FAU_CRP_EXT.1 Support for Compliance Reporting of Mobile Device Configuration

Hierarchical to: No other components.

Dependencies to: FTP_ITC.1 Inter-TSF Trusted Channel

FAU_CRP_EXT.1.1

The TSF shall provide [**assignment**: *type of interface*] to authorized entities over a channel that meets the secure channel requirements in FTP_ITC.1/INTER_XFER_IT. The provided information for each enrolled MD includes:

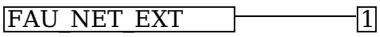
- The current version of the MD firmware or software
- The current version of the hardware model of the device
- The current version of installed applications
- List of MD configuration policies that are in place on the device (as defined in FMT_SMF.1.1/SERVER_CONF_AGENT)
- [**selection**: ***[assignment**: list of other available information about enrolled devices]** , *no other information*].

B.2.5.3 FAU_NET_EXT Network Reachability Review

Family Behavior

This family defines requirements for administrators to see network connectivity status.

Component Leveling



FAU_NET_EXT.1, Network Reachability Review, defines requirements for authorized administrators to read network connectivity status.

Management: FAU_NET_EXT.1

The following actions could be considered for the management functions in FMT.

- Query connectivity status.

Audit: FAU_NET_EXT.1

There are no auditable events foreseen.

FAU_NET_EXT.1 Network Reachability Review

Hierarchical to: No other components.

Dependencies to: FAU_ALT_EXT.2 Agent Alerts

FAU_NET_EXT.1.1

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

B.2.6 Security Management (FMT)

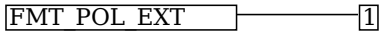
This PP defines the following extended components as part of the FMT class originally defined by CC Part 2:

B.2.6.1 FMT_POL_EXT Trusted Policy Update

Family Behavior

This family describes how to use digitally signed policies and updates by using private keys, and validating the policy is appropriate.

Component Leveling



FMT_POL_EXT.1, Trusted Policy Update, defines requirements for using digitally signed policies and policy updates.

Management: FMT_POL_EXT.1

There are no management activities foreseen.

Audit: FMT_POL_EXT.1

The following actions should be auditable if FAU_GEN security audit data generation is included in the PP or ST.

- Attempt to reuse enrollment data.
- Enrollment data.

FMT_POL_EXT.1 Trusted Policy Update

Hierarchical to: No other components.

Dependencies to: No dependencies.

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 [**selection**: *an X509 certificate, a public key provisioned to the agent*] trusted by the agent for policy verification.

FMT_POL_EXT.1.3

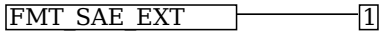
For each unique policy managed by the TSF, the TSF shall validate that the policy is appropriate for an agent using [**selection**: *client authentication via an X509 certificate representing the agent, a token issued to the agent and associated with a policy signing key uniquely associated to the policy*].

B.2.6.2 FMT_SAE_EXT Security Attribute Expiration

Family Behavior

This family defines the requirements for using expiration time for enrollment and denying enrollment if that time has passed.

Component Leveling



FMT_SAE_EXT.1, Security Attribute Expiration, defines requirements for the expiration time for enrollment authentication.

Management: FMT_SAE_EXT.1

The following action could be considered for the management functions in FMT.

- Configure the length of time the enrollment authenticator is valid.

Audit: FMT_SAE_EXT.1

The following actions should be auditable if FAU_GEN security audit data generation is included in the PP or ST.

- Enrollment attempted after expiration of authentication data.
- Identity of user.

FMT_SAE_EXT.1 Security Attribute Expiration

Hierarchical to:	No other components.
Dependencies to:	FIA_ENR_EXT.1 Enrollment of Mobile Device into Management FIA_UAU.4 Single-Use Authentication Mechanisms

FMT_SAE_EXT.1.1

The TSF shall be able to specify a configurable expiration time for enrollment authentication data.

FMT_SAE_EXT.1.2

The TSF shall be able to deny enrollment after the expiration time for the enrollment authentication data has passed.

B.2.7 Trusted Path/Channels (FTP)

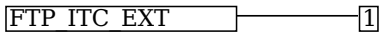
This PP defines the following extended components as part of the FTP class originally defined by CC Part 2:

B.2.7.1 FTP_ITC_EXT Trusted Channel

Family Behavior

The family defines requirements for communication channels between itself and other communication channels.

Component Leveling



FTP_ITC_EXT.1, Trusted Channel, defines requirements for providing logically distinct communication channels.

Management: FTP_ITC_EXT.1

There are no management activities foreseen.

Audit: FTP_ITC_EXT.1

There are no auditable events foreseen.

FTP_ITC_EXT.1 Trusted Channel

Hierarchical to:	No other components.
Dependencies to:	FTP_ITC.1 Inter-TSF Trusted Channel FTP_TRP.1 Trusted Path

FTP_ITC_EXT.1.1

The TSF shall provide a communication channel between itself and [**selection**:

- *an MDM agent that is internal to the TOE*
- *an MDM agent that is external to the TOE*
- *other components comprising the distributed TOE*

] that is logically distinct from other communication channels, as specified in [**assignment**: *inter-TSF trusted channel or internal TOE TSF data transfer type*]

Appendix C - Implicitly Satisfied Requirements

This appendix lists requirements that should be considered satisfied by products successfully evaluated against this PP. These requirements are not featured explicitly as SFRs and should not be included in the ST. They are not included as standalone SFRs because it would increase the time, cost, and complexity of evaluation. This approach is permitted by [CC] Part 1, 8.2 Dependencies between components.

This information benefits systems engineering activities which call for inclusion of particular security controls. Evaluation against the PP provides evidence that these controls are present and have been evaluated.

Table 8: Implicitly Satisfied Requirements

Requirement	Rationale for Satisfaction
FIA_UID.1 - Timing of Identification	FIA_UAU.1 has a dependency on FIA_UID.1 because authentication of an external entity requires that entity to first identify itself so that its identity can be validated by the authentication process. This SFR has not been defined in this PP-Module because authentication of a user implicitly requires them to be identified as well. There are two cases in which a user must be authenticated by the TSF per the application note in FIA_UAU.1: enrollment of a user's MD into management, and authentication of a user to the TOE's management interface. FIA_ENR_EXT.1 requires authentication of the user using a method such as a directory server, which implicitly expects that the user must identify themselves as well as provide a credential associated with the claimed identity. For management, FMT_SMR.1(1) requires the TSF to maintain an administrator role but it does not specify whether authentication to this role is role-based or identity-based. Therefore, the PP is not concerned with the specific mechanism by which an administrator presents their identity to the TSF, only that there is a mechanism to validate and authorize whatever information they do present.
FMT_MTD.1 - Management of TSF Data	FAU_SEL.1 has a dependency on FMT_MTD.1 because the configuration settings that determine what events are audited is considered to be TSF data. While FAU_SEL.1 determines the extent to which the TOE's audit function is configured, it relies on FMT_MTD.1 to determine the administrative roles that are permitted to manipulate this data. The PP allows FAU_SEL.1 to be implemented either by the TSF or by the TOE platform because the TOE may rely on the audit functionality provided by the OS it runs on. If this is configured entirely through the platform, the administrator does not necessarily need to be authenticated by the TOE to do this. Therefore, requiring FMT_MTD.1 does not make sense in this situation. If this is configured through the TOE, it can be implied from a review of FMT_SMR.1(1) that the 'MD user' role cannot perform this function as they lack the authority to manage the TSF. It is therefore understood that this function can be performed by the 'administrator' role or potentially by one or more roles specified by the ST author if the selection to specify additional roles is chosen. An additional SFR that explicitly identifies the roles permitted to manage this function is redundant in this context.
FPT_STM.1 - Reliable Time Stamps	The PP's iterations of FAU_GEN.1 as well as its cryptographic functionality have a dependency on FPT_STM.1 because audit data require accurate timestamps as well as some cryptographic operations, such as determining if a presented X.509 certificate is expired. The TOE is installed on a general-purpose computer or specialized network device that is assumed to have the ability to provide certain functions to the TOE as specified in A.MDM_SERVER_PLATFORM. This assumption explicitly lists 'reliable timestamps' as a function that the TSF is expected to have available to it.

Appendix D - Entropy Documentation and Assessment

This appendix describes the required supplementary information for each entropy source used by the TOE.

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

D.1 Design Description

Documentation shall include the design of each entropy source as a whole, including the interaction of all entropy source components. Any information that can be shared regarding the design should also be included for any third-party entropy sources that are included in the product.

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

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

If implemented, the design description shall include a description of how third-party applications can add entropy to the RBG. A description of any RBG state saving between power-off and power-on shall be included.

D.2 Entropy Justification

There should be a technical argument for where the unpredictability in the source comes from and why there is confidence in the entropy source delivering sufficient entropy for the uses made of the RBG output (by this particular TOE). This argument will include a description of the expected min-entropy rate (i.e., the minimum entropy (in bits) per bit or byte of source data) and explain that sufficient entropy is going into the TOE randomizer seeding process. This discussion will be part of a justification for why the entropy source can be relied on to produce bits with entropy.

The amount of information necessary to justify the expected min-entropy rate depends on the type of entropy source included in the product.

For developer provided entropy sources, in order to justify the min-entropy rate, it is expected that a large number of raw source bits will be collected, statistical tests will be performed, and the min-entropy rate determined from the statistical tests. While no particular statistical tests are required at this time, it is expected that some testing is necessary in order to determine the amount of min-entropy in each output.

For third-party provided entropy sources, in which the TOE vendor has limited access to the design and raw entropy data of the source, the documentation will indicate an estimate of the amount of min-entropy obtained from this third-party source. It is acceptable for the vendor to "assume" an amount of min-entropy, however, this assumption must be clearly stated in the documentation provided. In particular, the min-entropy estimate must be specified and the assumption included in the ST.

Regardless of type of entropy source, the justification will also include how the DRBG is initialized with the entropy stated in the ST, for example by verifying that the min-entropy rate is multiplied by the amount of source data used to seed the DRBG or that the rate of entropy expected based on the amount of source data is explicitly stated and compared to the statistical rate. If the amount of source data used to seed the DRBG is not clear or the calculated rate is not explicitly related to the seed, the documentation will not be considered complete.

The entropy justification shall not include any data added from any third-party application or from any state saving between restarts.

D.3 Operating Conditions

The entropy rate may be affected by conditions outside the control of the entropy source itself. For example, voltage, frequency, temperature, and elapsed time after power-on are just a few of the factors that may affect the operation of the entropy source. As such, documentation will also include the range of operating conditions under which the entropy source is expected to generate random data. Similarly, documentation shall describe the conditions under which the entropy source is no longer guaranteed to provide sufficient entropy. Methods used to detect failure or degradation of the source shall be included.

D.4 Health Testing

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

Appendix E - Evaluating Additional Components for a Distributed TOE

In the case of a distributed TOE the ST will identify an evaluated configuration that consists of a number of separate components chosen by the ST author, which collectively satisfy the requirements of the PP. This evaluated configuration need not be the minimum set of components that could possibly meet the PP (e.g., if the TOE is intended for large enterprise deployments then the evaluated configuration might include some redundancy in components in order to support expected connectivity and loads), but because this is the main configuration referred to in the ST and the evaluation, it is treated in this section as the minimum configuration of interest and is referred to here as the 'minimum configuration' as well as the 'evaluated configuration'.

In addition to the minimum configuration above, the ST may also identify (at the author's discretion, and subject to verification as described in this section) which TOE components can have instances added to an operational configuration without affecting the validity of the CC certification. The ST description may include constraints on how such components are added, including required or prohibited configurations of the components.

Extra instances of a TOE component must have the same hardware and software as the original component included in the evaluated configuration.

E.1 Evaluator Actions for Assessing the ST

TSS The evaluator shall examine the TSS to identify any extra instances of TOE components allowed in the ST and shall examine the description of how the additional components maintain the SFRs to confirm that it is consistent with the role that the component plays in the evaluated configuration. For example: the secure channels used by the extra component for intra-TOE communications (FPT_ITT) and external communications (FTP_ITC) must be consistent, the audit information generated by the extra component must be maintained, and the management of the extra component must be consistent with that used for the original instance of the component in the minimum configuration.

E.2 Evaluator Actions for Assessing the Guidance Documentation

Guidance w The evaluator shall examine the description of the extra instances of TOE components in the guidance documentation to confirm that they are consistent with those identified as allowed in the ST. This includes confirmation that the result of applying the guidance documentation to configure the extra component will leave the TOE in a state such that the claims for SFR support in each component are as described in the ST and therefore that all SFRs continue to be met when the extra components are present.

The evaluator shall examine the secure communications described for the extra components to confirm that they are the same as described for the components in the minimum configuration (additional connections between allowed extra components and the components in the minimum configuration are allowed of course).

E.3 Evaluator Actions for Testing the TOE

Tests The evaluator tests the TOE in the minimum configuration as defined in the ST (and the guidance documentation).

If the description of the use of extra components in the ST and guidance documentation identifies any difference in the SFRs allocated to a component, or the scope of the SFRs involved (e.g., if different selections apply to different instances of the component) then the evaluator tests these additional SFR cases that were not included in the minimum configuration.

In addition, the evaluator tests the following aspects for each extra component that is identified as allowed in the distributed TOE:

- Communications: the evaluator follows the guidance documentation to confirm, by testing, that any additional connections introduced with the extra component and not present in the minimum configuration are consistent with the requirements stated in the ST (e.g., with regard to protocols and ciphersuites used). An example of such an additional connection would be if a single instance of the component is present in the minimum configuration and adding a duplicate component then introduces an extra communication between the two instances. Another example might be if the use of the additional components necessitated the use of a connection to an external authentication server instead of using locally stored credentials.
- Audit: the evaluator confirms that the audit records from different instances of a component can be distinguished so that it is clear which instance generated the record.

- Management: if the extra component manages other components in the distributed TOE then the evaluator shall follow the guidance documentation to confirm that management via the extra component uses the same roles and role holders for administrators as for the component in the minimum configuration.

Appendix F - MDM Software Equivalency Guidelines

F.1 Introduction

The purpose of equivalence in PP-based evaluations is to find a balance between evaluation rigor and commercial practicability, to ensure that evaluations meet customer expectations while recognizing that there is little to be gained from requiring that every variation in a product or platform be fully tested. If a product is found to be compliant with a PP on one platform, then all equivalent products on equivalent platforms are also considered to be compliant with the PP.

A Vendor can make a claim of equivalence if the Vendor believes that a particular instance of their Product implements PP-specified security functionality in a way equivalent to the implementation of the same functionality on another instance of their Product on which the functionality was tested. The Product instances can differ in version number or feature level (model), or the instances may run on different platforms. Equivalency can be used to reduce the testing required across claimed evaluated configurations. It can also be used during Assurance Maintenance to reduce testing needed to add more evaluated configurations to a certification.

These equivalency guidelines do not replace Assurance Maintenance requirements or NIAP Policy #5 requirements for CAVP certificates. Nor may equivalency be used to leverage evaluations with expired certifications.

These Equivalency Guidelines represent a shift from complete testing of all product instances to more of a risk-based approach. Rather than require that every combination of product and platform be tested, these guidelines support an approach that recognizes that products are being used in a variety of environments, and often in cloud environments over where the vendor (and sometimes the customer) have little or no control over the underlying hardware. Developers should be responsible for the security functionality of their applications on the platforms they are developed for, whether that is an operating system, a virtual machine, or a software-based execution environment such as a container. But those platforms may themselves run within other environments, virtual machines or operating systems, that completely abstract away the underlying hardware from the application. The developer should not be held accountable for security functionality that is implemented by platform layers that are abstracted away. The implication is that not all security functionality will necessarily be tested for all platform layers down to the hardware for all evaluated configurations, especially for applications developed for software-based execution environments such as containers. For these cases, the balancing of evaluation rigor and commercial practicability tips in favor of practicability. Note that this does not affect the requirement that at least one product instance be fully tested on at least one platform with cryptography mapped to a CAVP certificate.

Equivalency has two aspects:

- ***Product Equivalence:*** Products may be considered equivalent if there are no differences between Product Models and Product Versions with respect to PP-specified security functionality.
- ***Platform Equivalence:*** Platforms may be considered equivalent if there are no significant differences in the services they provide to the Product, or in the way the platforms provide those services, with respect to PP-specified security functionality.

The equivalency determination is made in accordance with these guidelines by the Validator and Scheme using information provided by the evaluator or vendor.

F.2 Approach to Equivalency Analysis

There are two scenarios for performing equivalency analysis. One is when a product has been certified and the vendor wants to show that a later product should be considered certified due to equivalence with the earlier product. The other is when multiple product variants are going though evaluation together and the vendor would like to reduce the amount of testing that must be done. The basic rules for determining equivalence are the same in both cases. But there is one additional consideration that applies to equivalence with previously certified products. That is, the product with which equivalence is being claimed must have a valid certification in accordance with scheme rules and the Assurance Maintenance process must be followed. If a product’s certification has expired, then equivalence cannot be claimed with that product.

When performing equivalency analysis, the evaluator or vendor should first use the factors and guidelines for Product Model equivalence to determine the set of Product Models to be evaluated. In general, Product Models that do not differ in PP-specified security functionality are considered equivalent for purposes of evaluation against the APP PP.

If multiple revision levels of Product Models are to be evaluated, or to determine whether a revision of an evaluated product needs re-evaluation, the evaluator or vendor and validator should use the factors and guidelines for Product Version equivalence to analyze whether Product Versions are equivalent.

Having determined the set of Product Models and Versions to be evaluated, the next step is to determine the

set of Platforms that the Products must be tested on.

Each non-equivalent Product for which compliance is claimed must be fully tested on each non-equivalent platform for which compliance is claimed. For non-equivalent Products on equivalent platforms, only the differences that affect PP-specified security functionality must be tested for each product.

"*Differences in PP-Specified Security Functionality*" *Defined*
If PP-specified security functionality is implemented by the TOE, then differences in the actual implementation between versions or product models break equivalence for that feature. Likewise, if the TOE implements the functionality in one version or model and the functionality is implemented by the platform in another version or model, then equivalence is broken. If the functionality is implemented by the platform in multiple models or versions on equivalent platforms, then the functionality is considered different if the product invokes the platform differently to perform the function.

F.3 Specific Guidance for Determining Product Model Equivalence

Product Model equivalence attempts to determine whether different feature levels of the same product across a product line are equivalent for purposes of PP testing. For example, if a product has a "basic" edition and an "enterprise" edition, is it necessary to test both models? Or does testing one model provide sufficient assurance that both models are compliant?

Product models are considered equivalent if there are no differences that affect PP-specified security functionality, as indicated in Table 9.

Table 9: Determining Product Model Equivalence

Factor	Same or Different	Guidance
PP-Specified Functionality	Same	If the differences between Models affect only non-PP-specified functionality, then the Models are equivalent.
	Different	If PP-specified security functionality is affected by the differences between Models, then the Models are not equivalent and must be tested separately. It is necessary only to test the functionality affected by the software differences. If only differences are tested, then the differences must be enumerated, and for each difference the Vendor must provide an explanation of why each difference does or does not affect PP-specified functionality. If the Product Models are separately tested fully, then there is no need to document the differences.

F.4 Specific Guidance for Determining Product Version Equivalence

In cases of version equivalence, differences are expressed in terms of changes implemented in revisions of an evaluated Product. In general, versions are equivalent if the changes have no effect on any security-relevant claims about the TOE or assurance evidence. Non-security-relevant changes to TOE functionality or the addition of non-security-relevant functionality does not affect equivalence.

Table 10: Factors for Determining Product Version Equivalence

Factor	Same or Different	Guidance
Product Models	Different	Versions of different Product Models are not equivalent unless the Models are equivalent as defined in subsection 3.
PP-Specified Functionality	Same	If the differences affect only non-PP-specified functionality, then the Versions are equivalent.
	Different	If PP-specified security functionality is affected by the differences, then the Versions are not considered equivalent and must be tested separately. It is necessary only to test the functionality affected by the changes. If only the differences are tested, then for each difference the Vendor must provide an explanation of why the difference does or does not affect PP-specified functionality. If the Product Versions are separately tested fully, then there is no need to document the differences.

F.5 Specific Guidance for Determining Platform Equivalence

Platform equivalence is used to determine the platforms that equivalent versions of a Product must be tested on. Platform equivalence analysis done for one MDM cannot be applied to another MDM. Platform equivalence is not general, it is with respect to a particular MDM.

Product Equivalency analysis must already have been done and Products have been determined to be equivalent.

The platform can be hardware or virtual hardware, an operating system or similar entity, or a software execution environment such as a container. For purposes of determining equivalence for MDMs, we address each type of platform separately. In general, platform equivalence is based on differences in the interfaces between the TOE and Platform that are relevant to the implementation of PP-specified security functionality.

F.5.1 Platform Equivalence, Hardware and Virtual Hardware Platforms

If an MDM runs directly on hardware without an operating system, or directly on virtualized hardware without an operating system, then platform equivalence is based on processor architecture and instruction sets. In the case of virtualized hardware, it is the virtualized processor and architecture that are presented to the MDM that matters, not the physical hardware.

Platforms with different processor architectures and instruction sets are not equivalent. This is not likely to be an issue for equivalency analysis for MDMs since there is likely to be a different version of the MDM for different hardware environments. Equivalency analysis becomes important when comparing processors with the same architecture. Processors with the same architecture that have instruction sets that are subsets or supersets of each other are not disqualified from being equivalent for purposes of an MDM evaluation. If the MDM takes the same code paths when executing PP-specified security functionality on different processors of the same family, then the processors can be considered equivalent with respect to that MDM. For example, if an MDM follows one code path on platforms that support the AES-NI instruction and another on platforms that do not, then those two platforms are not equivalent with respect to that MDM functionality. But if the MDM follows the same code path whether or not the platform supports AES-NI, then the platforms are equivalent with respect to that functionality.

The platforms are equivalent with respect to the MDM if the platforms are equivalent with respect to all PP-specified security functionality.

Table 11: Factors for Determining Hardware and Virtual Hardware Platform Equivalence

Factor	Same, Different, or None	Guidance
Platform Architectures	Different	Platforms that present different processor architectures and instruction sets to the MDM are not equivalent.
PP-Specified Functionality	Same	For platforms with the same processor architecture, the platforms are equivalent with respect to the MDM if execution of all PP-specified security functionality follows the same code path on both platforms.

F.5.2 Platform Equivalence, OS Platforms

For MDMs that are built for and run on operating systems, platform equivalence is determined by the interfaces between the MDM and the operating system that are relevant to PP-specified security functionality. Generally, these are the processor interface, device interfaces, and OS APIs. The following factors applied in order:

Table 12: Factors for Determining OS or VS Platform Equivalence

Factor	Same, Different, or None	Guidance
Platform Architectures	Different	Platforms that run on different processor architectures and instruction sets are not equivalent.
Platform Vendors	Different	Platforms from different vendors are not equivalent.
Platform Versions	Different	Platforms from the same vendor with different major version numbers are not equivalent.
Platform	Different	Platforms from the same vendor and major version are not equivalent if there are

Interfaces		differences in device interfaces and OS APIs that are relevant to the way the platform provides PP-specified security functionality to the MDM.
Platform Interfaces	Same	Platforms from the same vendor and major version are equivalent if there are no differences in device interfaces and OS APIs that are relevant to the way the platform provides PP-specified security functionality to the MDM, or if the Platform does not provide such functionality to the MDM.

F.5.3 Software-based Execution Environment Platform Equivalence

If an MDM is built for and runs in a non-OS software-based execution environment, such as a Container or Java Runtime, then the below criteria must be used to determine platform equivalence. The key point is that the underlying hardware (virtual or physical) and OS is not relevant to platform equivalence. This allows MDMs to be tested and run on software-based execution environments on any hardware, as in cloud deployments.

Table 13: Factors for Software-based Execution Environment Platform Equivalence

Factor	Same, Different, or None	Guidance
Platform Type or Vendor	Different	Software-based execution environments that are substantially different or come from different vendors are not equivalent. For example, a java virtual machine is not the same as a container. A Docker container is not the same as a CoreOS container.
Platform Versions	Different	Execution environments that are otherwise equivalent are not equivalent if they have different major version numbers.
PP-Specified Security Functionality	Same	All other things being equal, execution environments are equivalent if there is no significant difference in the interfaces through which the environments provide PP-specified security functionality to MDMs.

F.6 Level of Specificity for Tested Configurations and Claimed Equivalent Configurations

In order to make equivalency determinations, the vendor and evaluator must agree on the equivalency claims. They must then provide the scheme with sufficient information about the TOE instances and platforms that were evaluated, and the TOE instances and platforms that are claimed to be equivalent.

The ST must describe all configurations evaluated down to processor manufacturer, model number, and microarchitecture version.

The information regarding claimed equivalent configurations depends on the platform that the MDM was developed for and runs on.

Bare-Metal MDMs

For MDM servers that run without an operating system on bare-metal or virtual bare-metal, the claimed configuration must describe the platform down to the specific processor manufacturer, model number, and microarchitecture version. The Vendor must describe the differences in the TOE with respect to PP-specified security functionality and how the TOE functions differently to leverage platform differences (e.g., instruction set extensions) in the tested configuration versus the claimed equivalent configuration.

Traditional MDMs

For MDMs that run with an operating system as their immediate platform, the claimed configuration must describe the platform down to the specific operating system version. If the platform is a virtualization system, then the claimed configuration must describe the platform down to the specific virtualization system version. The Vendor must describe the differences in the TOE with respect to PP-specified security functionality and how the TOE functions differently to leverage platform differences in the tested configuration versus the claimed equivalent configuration. Relevant platform differences could include instruction sets, device interfaces, and OS APIs invoked by the TOE to implement PP-specified security functionality.

Software-Based Execution Environments

For MDMs that run in a software-based execution environment such as a Java virtual machine or a Container, then the claimed configuration must describe the platform down to the specific version of the software

execution environment. The Vendor must describe the differences in the TOE with respect to PP-specified security functionality and how the TOE functions differently to leverage platform differences in the tested configuration versus the claimed equivalent configuration.

Appendix G - Use Case Templates

The following use case templates list those selections, assignments, and objective requirements that best support the use cases identified by this Protection Profile. Note that the templates assume that all SFRs listed in Section 5 are included in the ST, not just those listed in the templates. These templates and deviations from the template should be identified in the Security Target to assist customers with making risk-based purchasing decisions. Products that do not meet these templates are not precluded from use in the scenarios identified by this Protection Profile. Where selections for a particular requirement are not identified in a use case template, all available selections are equally applicable to the use case.

G.1 Enterprise-owned device for general-purpose enterprise use

Requirement	Action
FMT_SMF.1/SERVER_CONF_AGENT Function 32	Include in ST and assign GPS.
FMT_SMF.1/SERVER_CONF_AGENT Function 34	Include in ST. Assign personal hotspot connections (if feature exists).
FMT_SMF.1/SERVER_CONF_AGENT Function 47	Include in ST.
FMT_SMF.1/SERVER_CONF_AGENT Function 49	Include in ST and select "a. USB mass storage mode."
FMT_SMF.1/SERVER_CONF_AGENT Function 51	Include in ST. Select both options.

G.2 Enterprise-owned device for specialized, high-security use

Requirement	Action
FMT_SMF.1/SERVER_CONF_AGENT Function 15	Include in ST.
FMT_SMF.1/SERVER_CONF_AGENT Function 16	Include in ST.
FMT_SMF.1/SERVER_CONF_AGENT Function 31	Include in ST and select "no other method."
FMT_SMF.1/SERVER_CONF_AGENT Function 32	Include in ST.
FMT_SMF.1/SERVER_CONF_AGENT Function 33	Include in ST. Assign at least USB.
FMT_SMF.1/SERVER_CONF_AGENT Function 34	Include in ST. Assign all protocols where the TSF acts as a server.
FMT_SMF.1/SERVER_CONF_AGENT Function 36	Include in ST.
FMT_SMF.1/SERVER_CONF_AGENT Function 37	Include in ST.
FMT_SMF.1/SERVER_CONF_AGENT Function 40	Include in ST.
FMT_SMF.1/SERVER_CONF_AGENT Function 42	Include in ST.
FMT_SMF.1/SERVER_CONF_AGENT Function 47	Include in ST.
FMT_SMF.1/SERVER_CONF_AGENT Function 52	Include in ST.
FMT_SMF.1/SERVER_CONF_AGENT Function 54	Include in ST.

FMT_SMF.1/SERVER_CONF_AGENT Function c.1	Include in ST.
FMT_SMF.1/SERVER_CONF_AGENT Function c.2	Include in ST.
FCS_CKM.1.1	Select RSA with key size of 3072 or select ECC schemes.
FCS_CKM.2.1	Select "RSA schemes" or select "ECC schemes."
FCS_COP.1.1/CONF_ALG	Select 256 bits
FCS_COP.1.1/HASH_ALG	Select SHA-384
FCS_COP.1.1/SIGN_ALG	Select RSA with key size of 3072 or select ECC schemes.
FIA_X509_EXT.2.2	Select either "allow the administrator to choose…" or "not accept the certificate."
FCS_TLSC_EXT.1.1 (from TLS Package)	If included in ST, select "TLS 1.2."
FCS_TLSC_EXT.2.1 (from TLS Package)	If included in ST, select "secp384r1."

G.3 Personally-owned device for personal and enterprise use

Requirement	Action
FMT_SMF.1/SERVER_CONF_AGENT Function 13	Include in ST
FMT_SMF.1/SERVER_CONF_AGENT Function 14	Include in ST
FMT_SMF.1/SERVER_CONF_AGENT Function 21	Include in ST
FMT_SMF.1/SERVER_CONF_AGENT Function 22	Include in ST
FMT_SMF.1/SERVER_CONF_AGENT Function 30	Select "on a per-app basis," "on a per-group of application processes basis," or both
FMT_SMF.1/SERVER_CONF_AGENT Function 31	If included in ST, select "on a per-app basis," "on a per-group of application processes basis," or both
FMT_SMF.1/SERVER_CONF_AGENT Function 48	Include in ST
FMT_SMF.1/SERVER_CONF_AGENT Function 52	If included in ST, select "on a per-app basis," "on a per-group of application processes basis," or both
FMT_SMF.1/SERVER_CONF_SERVER Function f	Include in the ST

G.4 Personally-owned device for personal and limited enterprise use

At this time no requirements are recommended for this use case.

Appendix H - Acronyms

Table 14: Acronyms

Acronym	Meaning
API	API Application Programming Interface
Base-PP	Base Protection Profile
CC	Common Criteria
CEM	Common Evaluation Methodology
cPP	Collaborative Protection Profile
CSP	Critical Security Parameter
DEK	Data Encryption Key
EP	Extended Package
EST	Enrollment over Secure Transport
FP	Functional Package
KEK	Key Encryption Key
MAS	Mobile Application Store
MD	Managed Device
MDM	Mobile Device Management
OE	Operational Environment
OS	Operating System
PP	Protection Profile
PP-Configuration	Protection Profile Configuration
PP-Module	Protection Profile Module
REK	Root Encryption Key
SAR	Security Assurance Requirement
SFR	Security Functional Requirement
ST	Security Target
TOE	Target of Evaluation
TSF	TOE Security Functionality
TSFI	TSF Interface
TSS	TOE Summary Specification

Appendix I - Bibliography

Table 15: Bibliography

Identifier	Title
[CC]	Common Criteria for Information Technology Security Evaluation - • Part 1: Introduction and General Model, CCMB-2017-04-001, Version 3.1 Revision 5, April 2017. • Part 2: Security Functional Components, CCMB-2017-04-002, Version 3.1 Revision 5, April 2017. • Part 3: Security Assurance Components, CCMB-2017-04-003, Version 3.1 Revision 5, April 2017.
[APP PP]	Protection Profile for Application Software, Version 2.0, 2025-06-16
[CSA]	Computer Security Act of 1987, H.R. 145, June 11, 1987.
[MDF PP]	Protection Profile for Mobile Device Fundamentals, Version 4.0, 2022-09-12 Adjust date when finalized.
[MOD MDMA]	PP-Module for MDM agents, Version 2.0, 2019-04-25 Adjust date when finalized.
[MOD VPNC]	PP-Module for VPN Client, Version 3.0, 2024-06-24 Adjust date when finalized.
[OMB]	Reporting Incidents Involving Personally Identifiable Information and Incorporating the Cost for Security in Agency Information Technology Investments, OMB M-06-19, July 12, 2006.