www.GossamerSec.com

Assurance Activity Report for Samsung SDS EMM and EMM Agent for Android v2.2.5

Version 0.1
05/31/23

Prepared by:
Gossamer Security Solutions
Accredited Security Testing Laboratory – Common Criteria Testing
Columbia, MD 21045

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

Revision History

Revision	Date	Authors	Summary
Version 0.1	05/31/23	Gossamer	Initial draft

The TOE Evaluation was Sponsored by:

Samsung SDS Co. Ltd.

Samsung SDS Tower, 125,
Olympic-ro 35-gil, Songpa-gu,
Seoul, Korea 05510

Evaluation Personnel:

- Douglas Kalamus
- Rizheng Sun

Common Criteria Versions:

- Common Criteria for Information Technology Security Evaluation Part 1: Introduction, Version 3.1, Revision 5, April 2017
- Common Criteria for Information Technology Security Evaluation Part 2: Security functional components, Version 3.1, Revision 5, April 2017
- Common Criteria for Information Technology Security Evaluation Part 3: Security assurance components, Version 3.1, Revision 5, April 2017

Common Evaluation Methodology Versions:

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

Table of Contents

1. Introduction

This document presents evaluations results of Samsung SDS EMM and EMM Agent for Android version 2.2.5 MDMPP40/MDMA10/PKGTLS11 evaluation. This document contains a description of the assurance activities and associated results as performed by the evaluators.

1.1 Device Equivalence

The TOE server components are designed to work in Microsoft Windows Server 2016 and Server 2019. The evaluator set up two distributed working test configurations where each configuration included instances of all TOE components and involved two Windows Servers – one each of Server 2016 and Server 2019. The evaluators did not notice any difference in TOE behavior regardless of the Windows Server version." to "The TOE server components are designed to work in Microsoft Windows Server 2016 and Server 2019. The evaluators performed all of the reported testing on a Server 2016 platform with Java 1.8 and Intel(R) Xeon(R) CPU E3-1230. Windows Server 2019 is considered to be equivalent since it supports Java 1.8 and otherwise only common Windows APIs are used for primarily non-security-related functions with the majority of security-relevant support functions coming from Java. However, the evaluators did install the TOE on a Windows Server 2019 platform and also configured and exercised some security functions (e.g., IPsec connections) in order to gain further confidence of platform equivalence. The evaluators did not notice any difference in TOE behavior regardless of the Windows Server version.

The TOE Android agent is designed to work in a range of Android devices. For the purpose of testing, the evaluators selected one representative device from each of the applicable sets of evaluated devices on the NIAP PCL (i.e., one device from each evaluation). It is known that a single MDM test application was used in each MDF evaluation and a common Knox management API was identified for each of those evaluation. As such, by testing one sample device for each evaluation we have some assurance that the rest of the devices can be managed just like the device we specifically tested.

The iOS agents have been or are being evaluated outside the scope of this evaluation. Historically, only one agent is identified in each evaluation, so the evaluators chose one sample from each identifiable evaluation. Note that iOS15 is currently on the NIAP PCL and iOS16 is currently on the NIAP in-evaluation list, so two devices were used for testing to represent the corresponding agents.

1.2 CAVP Certificates

The EMM components use RSA BSAFE Crypto-J version 6.3 which performs all cryptographic operations.

SFR	NIST Standard	RSA Crypto-J 6.3 CAVP Cert #
FCS_CKM.1 – Key Generation *RSA, FFC and ECDSA key gen*		
RSA	FIPS 186-4	A3218

186-4: Key(gen) – 2048 bit		
ECDSA 186-4: Key(gen) P256, P-384	FIPS 186-4	A3218
DSA FFC 186-4: Key(gen) – 2048 bit	FIPS 186-4	A3218
FCS_CKM.2 – Key Establishment		
KAS ECC KAS FFC	NIST SP 800-56A	A3218
RSA-based key establishment schemes	RFC 8017	Vendor Affirmed
FCS_COP.1(*)		
AES 128/256 CBC, GCM	FIPS 197, SP 800-38A/D	A3218
RSA SigGen(2048), SigVer(2048)	FIPS 186-4	A3218
ECDSA SigGen/SigVer (P-256, P-384)	FIPS 186-4	A3218
SHA-256/384/512	FIPS 180-4	A3218
HMAC SHA-256/384	FIPS 198-1 & 180-4	A3218
FCS_RBG_EXT.1		
DRBG HMAC_DRBG (SHA-256)		A3218

2. Protection Profile SFR Assurance Activities

This section of the AAR identifies each of the assurance activities included in the claimed Protection Profile and describes the findings in each case.

The following evidence was used to complete the Assurance Activities:

AAR v0.1

- Samsung SDS EMM and EMM Agent for Android version 2.2.5 Security Target, Version 0.92, 5/26/2023 **ST**
- Samsung SDS EMM Administrator's Guide, Version 2.2.5, January 2023 [**Admin Guide**]
- Samsung SDS EMM Installation Guide, Version 2.2.5, January 2023 [**Install Guide**]
- Samsung SDS EMM Configuration Guide for IPsec settings in Microsoft Windows Server 2016 for Common Criteria Evaluation, Version 2.2.5, 27 January 2023 [**IPsec Guide**]

2.1 Security audit (FAU)

2.1.1 Server Alerts (MDMPP40:FAU_ALT_EXT.1)

2.1.1.1 MDMPP40:FAU_ALT_EXT.1.1

TSS Assurance Activities: None Defined

Guidance Assurance Activities: None Defined

Testing Assurance Activities: None Defined

Component TSS Assurance Activities: 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.

Section 6.1 of ST indicates that the EMM Server alerts administrators by displaying a "notifications" tab containing alerts for the administrator. Currently, the EMM Server displays alerts received from the MDM agent for changes in enrollment status (i.e., successful un/enrollment of devices) and a failure of an Agent to apply policies. The EMM Server enforces no limit on the maximum number of queued messages, and queues messages for the administrator in the notifications tab.

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

Section 12 of [Admin] the subsection entitled "Managing alerts" describes both how to view alerts as well as how to configure alerts.

Component Testing Assurance Activities: For each MDM Agent/platform listed as supported in the ST:

Test 1: The evaluator shall enroll a device and ensure that theMDM 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 FAU_GEN.1.1(1) change in enrollment status.

Test 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 theMDM 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.

Test 3: (Conditional) The evaluator shall trigger each of the events listed and ensure that the MDM Server alerts the administrator.

Test 1: The evaluator configured alerts and proceeded to enroll and unenroll a device. The evaluator observed that the alerts were properly generated in response to the action performed.

Test 2: The evaluator configured policies as defined below and ensured that the failure condition was detected and an alert was sent to the administrator using the configured methods.

1. Configure a policy that is not supported by the platform and verify that the MDM server alerts the administrator.

2. Manually modify a policy prior to transmission to the device such that it contains a valid configuration variable with an invalid value, then verify that the MDM server alerts the administrator.

Test 3 was not performed as there were no additional events beyond those tested by the preceding test cases.

2.1.2 Agent Alerts (MDMA10:FAU_ALT_EXT.2)

2.1.2.1 MDMA10:FAU_ALT_EXT.2.1

TSS Assurance Activities: None Defined

Guidance Assurance Activities: None Defined

Testing Assurance Activities: None Defined

2.1.2.2 MDMA10:FAU_ALT_EXT.2.2

TSS Assurance Activities: None Defined

Guidance Assurance Activities: None Defined

Testing Assurance Activities: None Defined

Component TSS Assurance Activities: The evaluator shall examine the TSS and verify that it describes how the alerts are implemented.

The evaluator shall examine the TSS and verify that it describes how the candidate policy updates are obtained and the actions that take place for successful (policy update installed) and unsuccessful (policy update not installed) cases. The software components that are performing the processing must also be identified in the TSS and verified by the evaluator.

The evaluator also ensures that the TSS describes how reachability events are implemented, and if configurable are selected in FMT_SMF_EXT.4.2. The evaluator verifies that this description clearly indicates who (MDM Agent or MDM Server) initiates reachability events.

The evaluator shall ensure that the TSS describes under what circumstances, if any, the alert may not be generated (e.g., the device is powered off or disconnected from the trusted channel), how alerts are queued, and the maximum amount of storage for queued messages.

Section 6.1 of ST indicates that the EMM Server displays alerts when policies are applied; reachability status (which could be triggered by the Server or Client); change in enrollment state; and failure to install or update an application from the MAS Server. This includes describing that candidate policies are received by the EMM Agent, sent by the enrolled EMM, via mutually authenticated TLS. This section describes that when received the candidate policy is checked according to MDMA10:FMT_POL_EXT.2 where an alert is sent if the policy is not accepted (i.e., fails its signature check). If the check succeeds, the EMM Agent checks each policy setting and applies the settings that are valid for the given device using available management APIs. This section describes that the EMM Agent sends an alert after applying policies including identifying any invalid settings.

Section 6.1 of ST also indicates that an administrator can query a device to obtain its current status. If the specified device does not have network connectivity, the EMM queues the query and delivers it when the device next contacts the Server. This section describes that reachability events can be initiated by the EMM Agent when it sends any alerts or other messages to the EMM and alternately can be initiated by the EMM where it can make requests of the EMM Agent. The reachability status on the EMM is based on any secure communication with the EMM Agent.

Section 6.1 of ST indicates that alerts are sent across the mutually authenticated TLS channel to the EMM available once a device is enrolled and if that channel is not available any alerts are queued and forwarded when the

channel is re-established. This section further describes that if the communication channel used for alerts is not available (e.g., Airplane mode or other network discontinuities), the EMM Agent will cache alerts in its local file storage and will send those alerts once connectivity is restored. The local file storage is limited only by the space available on the mobile device.

Component Guidance Assurance Activities: None Defined

Component Testing Assurance Activities: Test 1: The evaluator shall perform a policy update from the test environment MDM server. The evaluator shall verify the MDM Agent accepts the update, makes the configured changes, and reports the success of the policy update back to the MDM Server.

Test 2: The evaluator shall perform each of the actions listed in FAU_ALT_EXT.2.1 and verify that the alert does in fact reach the MDM Server.

Test 3: The evaluator shall configure the MDM Agent to perform a network reachability test, both with and without such connectivity and ensure that results reflect each.

Test 4: The evaluator shall remove network connectivity from the MDM Agent and generate an alert/event as defined in FAU_ALT_EXT.2.1. The evaluator shall restore network connectivity to the MDM Agent and verify that the alert generated while the TOE was disconnected is sent by the MDM Agent upon re-establishment of the connectivity.

Test 1: The evaluator used the EMM console to push a device policy and observed that the device installed the policy, acknowledged the policy to the EMM and reported success within the audit logs.

Test 2: The evaluator performed each of the following actions from the EMM and verified that the EMM generated alerts:

- successful application of policies to a mobile device
- receiving and generating periodic reachability events
- change in enrollment state
- failure to install an application from the MAS Server
- failure to update an application from the MAS Server

Test 3 & 4: The evaluator confirmed that network connectively on a target device with network connectivity functioned as expected. The evaluator observed that connectivity status was updated following a device command being sent from the EMM. The evaluator then enabled airplane mode for approximately 5 minutesand issued a device command. After about 5 minutes the evaluator disabled airplane mode and observed that the "Last Seen" time for the mobile device updated itself without further action by the evaluator

2.1.3 Audit Data Generation (MDMPP40:FAU_GEN.1(1))

2.1.3.1 MDMPP40:FAU_GEN.1.1(1)

TSS Assurance Activities: 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 FAU_GEN.1.2(1) 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).

Section 6.1 of ST provides a complete list of audit events and their contents by referencing Table 2 in the associated SFR. The description states the TOE (EMM) creates the required audit events.

Guidance Assurance Activities: 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.

Section 13 of [Admin] the subsection entitled "List of Audit Events" contains information describing the audit records that are generated by the TOE. The content and method of viewing audit records is described in Section 13 of [Admin] in the subsection entitled "Viewing Audit logs". The evaluator mapped each audit record during testing to Admin Guide and ensured all were collected and the details matched.

Testing Assurance Activities: The evaluator shall test the TOEs ability to correctly generate audit records by having the TOE generate audit records 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 records 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 records are generated as expected.

The evaluator constructed a list of required audit events based on the TOE Security Target ST. The evaluator then either identified events generated when performing a corresponding function in another test or performed operations directly (e.g., log in and log out) to cause each required audit event to be generated. The evaluator captured screen shots and/or textual content that is included in the proprietary detailed test report.

2.1.3.2 MDMPP40:FAU_GEN.1.2(1)

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

MDMPP40: The evaluator shall check the TSS and ensure that it provides a format for audit records. Each audit record format type must be covered, along with a brief description of each field.

Section 6.1 of ST indicates the content of audit events including the additional content specified in the table 2 with the requirement. Audit events include Date and Time, an Admin ID and Mobile IDs (if applicable), a Client IP (indicating the subject), an Event Category (type), an Event (indicates success or failure), and a Severity along with the additional information in column 3 of the Auditable Events table.

Guidance Assurance Activities: The evaluator shall check the administrative guide and ensure that it provides a format for audit records. Each audit record 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(1).

MDMPP40: The evaluator shall check the administrative guide and ensure that it provides a format for audit records. Each audit record 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(1).

The section entitled “Viewing audit logs” in Section 12 of [Admin] includes a description of the fields that define the format of an audit record. Additionally, the section entitled “Audit log fields in an exported log file” in section 13 of [Admin] also describes the various fields that define the format of an audit record. Refer to the analysis in MDMPP40:FAU_GEN.1(1).1 for information about required audit events and content

Testing Assurance Activities: When verifying the test results from FAU_GEN.1.1(1), the evaluator shall ensure the audit records generated during testing match the format specified in the administrative guide, and that the fields in each audit record 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 records are generated as expected.

The Auditable Events table includes optional, selection-based and objective requirements. The auditing of these requirements are only required if the requirement is included in the ST.

(man) - mandatory requirement

(sel) - selection-based requirement

(obj) - objective requirement

(opt) - optional requirement

MDMPP40: When verifying the test results from FAU_GEN.1.1(1), the evaluator shall ensure the audit records generated during testing match the format specified in the administrative guide, and that the fields in each audit record have the proper entries.

Note that the testing here can be accomplished in conjunction with the testing of 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 records are generated as expected.

The Auditable Events table includes optional, selection-based and objective requirements. The auditing of these requirements are only required if the requirement is included in the ST.

(man) - mandatory requirement

(sel) - selection-based requirement

(obj) - objective requirement

(opt) - optional requirement

While performing the FAU_GEN.1(1).1 tests, the evaluators collected and confirmed that the required audit record content was present in each audit record.

Component TSS Assurance Activities: None Defined

Component Guidance Assurance Activities: None Defined

Component Testing Assurance Activities: None Defined

2.1.4 Audit Data Generation (MDMA10:FAU_GEN.1(2))

2.1.4.1 MDMA10:FAU_GEN.1.1(2)

TSS Assurance Activities: None Defined

Guidance Assurance Activities: None Defined

Testing Assurance Activities: None Defined

2.1.4.2 MDMA10:FAU_GEN.1.2(2)

TSS Assurance Activities: None Defined

Guidance Assurance Activities: None Defined

Testing Assurance Activities: None Defined

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

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 Agent; nonetheless, that mechanism will be identified in the TSS as part of this evaluation activity).

Section 6.1 of ST indicates the content of audit events including the additional content specified in the table 3 with the requirement. Audit events include Date and Time, an Admin ID and Mobile IDs (if applicable), a Client IP (indicating the subject), an Event Category (type), an Event (indicates success or failure), and a Severity along with the additional information in column 3 of the Auditable Events table.

Component Guidance Assurance Activities: None Defined

Component Testing Assurance Activities: The evaluator shall use the TOE to perform the auditable events defined in the Auditable Events table in FAU_GEN.1.1(2) and observe that accurate audit records are generated with contents and formatting consistent with those described in the TSS. Note that this testing can be accomplished in conjunction with the testing of the security mechanisms directly.

The evaluator constructed a list of required audit events based on the TOE Security Target ST and MDMA10:FAU_GEN.1(2). The evaluator then either identified events generated when performing a corresponding function in another test or performed operations directly (e.g., log in and log out) to cause each required audit event to be generated. The evaluator captured screen shots and/or textual content that is included in the proprietary detailed test report. While performing these actions, the evaluators collected and confirmed that the required audit record content was present in each audit record.

2.1.5 Audit Generation (MAS Server) (MDMPP40:FAU_GEN.1(2))

2.1.5.1 MDMPP40:FAU_GEN.1.1(2)

TSS Assurance Activities: The evaluator shall check the TSS and ensure that it provides a format for audit records.

Section 6.1 of the ST states the TOE generates the required MAS audit events:

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

These match the SFR. Section 6.1 also explains that for each event in the TOE's audit log includes Log Data and Time, an Admin ID and Mobile IDs (if applicable), a Client IP (indicating the subject), an Event Category (type), an Event (indicates success or failure), and a Severity.

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

In the architecture of this product, the MAS server is a feature embedded within the EMM and is not a distinct architectural component. The audits identified by FAU_GEN.1(2) are found in Section 13 of [Admin] in the subsection entitled "List of Audit Events."

Testing Assurance Activities: The evaluator shall verify that when an application or update push fails, that the audit records generated match the format specified in the guidance and that the fields in each audit record have the proper entries.

The evaluator constructed a list of required audit events based on the TOE Security Target ST and FAU_GEN.1(2). The evaluator then either identified events generated when performing a corresponding function in another test or performed operations directly (e.g., log in and log out) to cause each required audit event to be generated. The evaluator captured screen shots and/or textual content that is included in the proprietary detailed test report.

2.1.5.2 MDMPP40:FAU_GEN.1.2(2)

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

In the architecture of this product, the MAS server is a feature embedded within the EMM and is not a distinct architectural component. The audits for the MAS server were described in the table of auditable events found in Table 2 in a manner identical to the audits for the EMM

Guidance Assurance Activities: The evaluator shall check the administrative guide and ensure that it provides a format for audit records. Each audit record 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(2).

In the architecture of this product, the MAS server is a feature embedded within the EMM and is not a distinct architectural component. The audits identified by [MDMPP40] for the MAS server were described in the guidance as described above in the Guidance Assurance Activity for FAU_GEN.1(1). As stated there the section entitled "Viewing audit logs" in Section 12 of [Admin] includes a description of the fields that define the format of an audit record. Additionally, the section entitled "Audit log fields in an exported log file" in section 13 of [Admin] also describes the various fields that define the format of an audit record.

Testing Assurance Activities: When verifying the test results from FAU_GEN.1.1(2), the evaluator shall ensure the audit records generated during testing match the format specified in the administrative guide, and that the fields in each audit record 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 records are generated as expected.

While performing the FAU_GEN.1(2).1 tests, the evaluators collected and confirmed that the required audit record content was present in each audit record.

Component TSS Assurance Activities: None Defined

Component Guidance Assurance Activities: None Defined

Component Testing Assurance Activities: None Defined

2.1.6 Network Reachability Review (MDMPP40:FAU_NET_EXT.1)

2.1.6.1 MDMPP40:FAU_NET_EXT.1.1

TSS Assurance Activities: None Defined

Guidance Assurance Activities: None Defined

Testing Assurance Activities: None Defined

Component TSS Assurance Activities: The evaluator ensures that the TSS describes how reachability events are implemented, for each supported mobile platform. The evaluator verifies that this description clearly indicates who (MDM Agent or MDM Server) initiates reachability events.

Section 6.1 of ST indicates that reachability events (device synch) normally occur periodically, where an administrator configured the period. Device synchs can also be initiated by an administrator using the EMM Server web interface to cause an immediate check-in to ensure or determine the current connectivity status. This behavior is an EMM Server function which is the same regardless of the type of mobile platform being targeted.

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

The agent periodically synchronizes with the server. By viewing "Detailed Information" about a device, the administrator can determine the most recent connection time for the device, the last time the policy was synced, and the last time the application list was updated. The TOE automatically updates the "Last Seen" times upon each contact with the device. Refer to the subsection heading "Viewing the device details" in section 4 of [Admin] for an explanation of the information the EMM collects about devices.

The administrator can send commands to an agent requesting an immediate sync. The instructions to send this command can be found in section 4 of [Admin] under the heading "Viewing the device list." The specific commands available are each described in Section 4 of [Admin] in the subsection entitled "List of Device Commands." The specific commands of interest are "Sync Device Information" and "Sync Installed App List.".

Component Testing Assurance Activities: For each MDM Agent/platform listed as supported in the ST: The evaluator shall configure the MDM Agent/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.

The evaluator configured network connectivity between the mobile devices (both Android and iOS) and the MDM server and attempted to use the server to check-in with the MD Agent. The evaluator observed on the server that the device status update was successful. The evaluator then changed the network connectivity such that the mobile device cannot reach the MDM server. With the network between the mobile device and the MDM server disrupted, the evaluator attempted to use the agent to do a status update to the MDM server. The evaluator observed on the server that the device status update was NOT successful. Finally, the evaluator re-established connectivity and observed that the status update occurred.

2.1.7 Audit Review (MDMPP40:FAU_SAR.1)

2.1.7.1 MDMPP40:FAU_SAR.1.1

TSS Assurance Activities: 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 TOE selection was implement functionality.

Guidance Assurance Activities: None Defined

Testing Assurance Activities: None Defined

2.1.7.2 MDMPP40:FAU_SAR.1.2

TSS Assurance Activities: 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 TOE selection was implement functionality.

Guidance Assurance Activities: The evaluator shall check the AGD guidance and ensure that it describes how the administrator accesses the audit data and describes the format of the audit record.

Section 11 of [Admin] the subsection entitled "Viewing Audits" describes the interface an administrator uses to view audit logs generated by the agent and server. while the heading "Viewing device log" describes the interface an administrator uses to view device logs generated by the agent. Section 12 of [Admin] in the subsection entitled "List of Audit Events" indicates that audit events generated by EMM, Push and App Tunnel servers are recorded to a log file in the platform server, and are viewed through remote access to the server. For these types of logs, the administrator can specify a date such that audits from that date are displayed. For device logs, the administrator can filter the displayed device logs based upon User ID or Mobile ID. For audit logs, the administrator can filter the displayed audit records based upon Admin ID, Mobile ID or Event. Additionally, the administrator can use a search box to attempt to locate specific strings within the audit records in the audit log.

Testing Assurance Activities: The evaluator shall attempt to view the audit record as the authorized administrator and verify that the action succeeds. The evaluator shall ensure the audit records generated during testing match the format specified in the administrative guide.

The evaluator viewed audit records through the EMM Console Interface and displayed a file in an Excel file format. Audit data for TLS connections associated with the MDM server's HTTPS interface are provided by the TOE as a download of low-level log text files.

Component TSS Assurance Activities: None Defined

Component Guidance Assurance Activities: None Defined

Component Testing Assurance Activities: None Defined

2.1.8 Security Audit Event Selection (MDMPP40:FAU_SEL.1)

2.1.8.1 MDMPP40:FAU_SEL.1.1

TSS Assurance Activities: None Defined

Guidance Assurance Activities: None Defined

Testing Assurance Activities: None Defined

Component TSS Assurance Activities: 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 TOE selection was implement functionality.

Component Guidance Assurance Activities: 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 preselection as well as explain the syntax (if present) for multi-value pre-selection. The administrative guidance shall also identify those audit records that are always recorded, regardless of the selection criteria currently being enforced.

Section 6 in [Admin] in the subsection entitled "Configuring Android Legacy Policies" and "Logging", is a policy setting named "Save logs".

This policy setting allows device auditing to be enabled or disabled, while additional settings within this same subsection control the level of auditing and storage constraints. Section 12 in [Admin], in the subsection entitled "Setting audit events", explains how to choose the specific audit events which the TOE is expected to collect (including the device events).

Component Testing Assurance Activities: The evaluator shall also perform the following tests:

Test 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 2: [conditional] If the TSF supports specification of more complex audit preselection 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.

Test 1: The evaluator selected specific "Audit Events" to be collected on devices and performed actions on tested devices to cause those events to be generated. The evaluator observed the configured audit records in the device's audit data. The evaluator then disabled the same "Audit Events" and performed the same actions on tested devices that would cause those events to be generated. The evaluator observed the configured audit records were not in the device's audit data.

Test 2: The TOE does not support complex combinations of pre-selection criteria.

2.1.9 Security Audit Event Selection (MDMA10:FAU_SEL.1(2))

2.1.9.1 MDMA10:FAU_SEL.1.1(2)

TSS Assurance Activities: None Defined

Guidance Assurance Activities: None Defined

Testing Assurance Activities: None Defined

Component TSS Assurance Activities: If 'invoke platform-provided functionality' is selected, the evaluator shall examine the TSS of the ST 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 Agent; nonetheless, that mechanism will be identified in the TSS as part of this evaluation activity).

The TOE selection was implement functionality.

Component Guidance Assurance Activities: The evaluator shall examine the operational guidance to determine that it contains instructions on how to define the set of auditable events as well as explains the syntax for multi-value selection (if applicable). The evaluator shall also verify that the operational guidance shall identify those audit records that are always recorded, regardless of the selection criteria currently being enforced.

Section 6 in [Admin] in the subsection entitled "Configuring Android Legacy Policies" and "Logging", is a policy setting named "Save logs".

This policy setting allows device auditing to be enabled or disabled, while additional settings within this same subsection control the level of auditing and storage constraints. Section 12 in [Admin], in the subsection entitled

"Setting audit events", explains how to choose the specific audit events which the TOE is expected to collect (including the device events).

Component Testing Assurance Activities: Test 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 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.

Test 1: The evaluator selected specific "Audit Events" to be collected on devices and performed actions on tested devices to cause those events to be generated. The evaluator observed the configured audit records in the device's audit data. The evaluator then disabled the same "Audit Events" and performed the same actions on tested devices that would cause those events to be generated. The evaluator observed the configured audit records were not in the device's audit data.

Test 2: The TOE does not support complex combinations of pre-selection criteria.

2.1.10 External Trail Storage (MDMPP40:FAU_STG_EXT.1)

2.1.10.1 MDMPP40:FAU_STG_EXT.1.1

TSS Assurance Activities: None Defined

Guidance Assurance Activities: None Defined

Testing Assurance Activities: None Defined

Component TSS Assurance Activities: 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.

Section 6.1 of the ST state the EMM always stores audit records locally in flat files stored on the TOE platform file system. The EMM provides administrators the capability to securely export EMM Console audit data through the EMM Server's administrative interface (WebUI) that is protected by an HTTPS trusted channel or alternately to a configured SYSLOG server that is protected using IPsec implemented by the underlying platform and configured according to available guidance documents. Windows log files generated by the server component's underlying Microsoft Platform can be locally access on the underlying platform or remotely using a RDP (Remote Desktop Protocol) or IPsec secured connection.

Component Guidance Assurance Activities: 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.

Section 12 of [Admin] the subsection entitled "Viewing Audit logs" explains that audit data resides within the product and is visible through the Service Overview > Log and Event > Audit Log. The subsection "Exporting audit logs to an Excel file" within section 12 explains that to export audit data, the administrator must use the "Export" button on the Device Logs and Audit logs screens. The guidance explains that device and audit log data is exported to the user's computer through the HTTPS/TLS or IPsec protected connection established when the administrator logged into the EMM Server. The subsection "Exporting audit logs to an Excel file" explains that export does not delete the device and audit logs from the EMM Server. The guidance explains that the EMM stores its audit records in storage provided by the environment, and deletion of audit record must be performed using operations in the environment.

Audit data is transferred from the EMM Agent to the EMM using the Internal TOE TSF data transfer communication channels, and thus is protected using TLS (See FPT_ITT.1.(2)). No additional configuration is necessary see FPT_ITT.1(2) for guidance references supporting configuration of the Internal TOE TSF data transfer communication channels.

Component Testing Assurance Activities: 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 1: The evaluator shall establish a session between theTOE 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.

The evaluator exported audit data from the MDM server and obtained a packet capture during the export activity. A review of the packet capture showed that the data transferred is not plaintext. A review of the audit data following the export showed that the audit data for the last record generated was properly recorded in the exported data.

2.1.11 Audit Event Storage (MDMPP40:FAU_STG_EXT.2)

2.1.11.1 MDMPP40:FAU_STG_EXT.2.1

TSS Assurance Activities: None Defined

Guidance Assurance Activities: None Defined

Testing Assurance Activities: None Defined

Component TSS Assurance Activities: 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).

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

Section 6.1 of the ST states the EMM secures its audit records by storing them in flat files protected by file access permissions enforced by the TOE Platform (Windows operating system) that preclude the ability to modify, insert, or delete a record (notwithstanding users with administrative rights on the TOE platform). Furthermore, the EMM only allows authenticated administrators access to display audit records, but provides no capability for an administrator to change those records.

The EMM Agent stores its audit logs within its local files and transmits those to the EMM via the MDMPP40:FPT_ITT.1(2) channel using TLS when requested by the enrolled EMM.

Component Guidance Assurance Activities: None Defined

Component Testing Assurance Activities: The evaluator shall perform the following tests:

Test 1: The evaluator shall access the audit trail as an unauthorized user and attempt to modify and delete the audit records. The evaluator shall verify that these attempts fail.

Test 2: The evaluator shall access the audit trail as an authorized user and attempt to modify and delete the audit records. 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.

Since only authorized administrators can log into the EMM Server and no interface to audit records exist outside of the EMM console and from the underlying server platform, the evaluator inspected the EMM console and found it is not possible for an unauthorized user to modify audit records.

2.2 Cryptographic support (FCS)

2.2.1 Cryptographic Key Generation (MDMPP40:FCS_CKM.1)

2.2.1.1 MDMPP40:FCS_CKM.1.1

TSS Assurance Activities: None Defined

Guidance Assurance Activities: None Defined

Testing Assurance Activities: None Defined

Component TSS Assurance Activities: 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.

Section 6.2 of the ST states that Each EMM server component uses its RSA BSAFE Crypto-J cryptographic module to generate asymmetric RSA, DSA, and ECDSA keys as part of key establishment. For authentication, the EMM allows the administrator to import RSA and ECDSA certificates into each EMM server component, as the components only generate keys during TLS key exchange.

The EMM Agent relies upon the EMM for generation of RSA and ECDSA keypairs. During the EMM Agent's enrollment process, the EMM Agent relies upon the EMM to generate either an RSA or ECDSA keypair on behalf of the Agent, to send a CSR request to the CA, and then returns (through an HTTPS protected session) the newly generated private key and issued certificate (as a PFX/PKCS12 file) to the Agent

Component Guidance Assurance Activities: The evaluator shall verify that the AGD 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.

The TOE allows an administrator to configure only the TLS ciphers used to protect communication between the Push server and the AppTunnel server. All other TLS communication cannot be configured by an administrator and uses default values.

Component Testing Assurance Activities: Key Generation for FIPS PUB 186-4 RSA Schemes: FIPS

Key Generation for Elliptic Curve Cryptography (ECC): FIPS 186-4 ECC Key Generation Test:

For each supported NIST curve, i.e., P-256, P-384 and P-521, the evaluator shall require the implementation under test (IUT) to generate 10 private/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.

Key Generation for Elliptic Curve Cryptography (ECC): FIPS 186-4 Public Key Verification (PKV) Test:

For each supported NIST curve, i.e., P-256, P-384 and P-521, the evaluator shall generate 10 private/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/FAIL values.

Key Generation for Finite-Field Cryptography (FFC): FIPS

Diffie-Hellman Group 14 and FFC Schemes using 'safe-prime' groups:

Testing for FFC Schemes using Diffie-Hellman group 14 and/or 'safe-prime' groups is done as part of testing in FCS_CKM.2.1.

See Section 1.2 for a listing of applicable CAVP certificates.

2.2.2 Cryptographic Key Establishment (MDMPP40:FCS_CKM.2)

2.2.2.1 MDMPP40:FCS_CKM.2.1

TSS Assurance Activities: None Defined

Guidance Assurance Activities: None Defined

Testing Assurance Activities: None Defined

Component TSS Assurance Activities: 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.

If Diffie-Hellman group 14 is selected from FCS_CKM.2.1, the TSS shall describe how the implementation meets RFC 3526 Section 3.

Section 6.2 of ST indicates that the TOE provides asymmetric key generation for key establishment within TLS and HTTPS. This function is provided by the EMM's RSA Crypto-J version 6.2, which has undergone CAVP validation.

Component Guidance Assurance Activities: The evaluator shall verify that the AGD guidance instructs the administrator how to configure the TOE to use the selected key establishment scheme(s).

The TOE allows an administrator to configure only the TLS ciphers used to protect communication between the Push server and the AppTunnel server. All other TLS communication cannot be configured by an administrator and uses default values that are compliant with the ciphersuites listed in FCS_TLSC_EXT.1.

Component Testing Assurance Activities: 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: FIPS

RSA-based key establishment:

The evaluator shall verify the correctness of the TSF's implementation of RSAESPKCS1-v1_5 by using a known good implementation for each protocol selected in FTP_TRP.1(1), FTP_TRP.1(2), FTP_TRP.1(3), FTP_ITC.1(1), FTP_ITC.1(2), FPT_ITT.1(1), and FPT_ITT.1(2) that uses RSAES-PKCS1-v1_5.

Diffie-Hellman Group 14:

The evaluator shall verify the correctness of the TSF's implementation of Diffie-Hellman group 14 by using a known good implementation for each protocol selected in FTP_TRP.1(1), FTP_TRP.1(2), FTP_TRP.1(3), FTP_ITC.1(1), FTP_ITC.1(2), FPT_ITT.1(1), and FPT_ITT.1(2) that uses Diffie-Hellman Group 14.

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(1), FTP_TRP.1(2), FTP_TRP.1(3), FTP_ITC.1(1), FTP_ITC.1(2), FPT_ITT.1(1), and FPT_ITT.1(2) that uses 'safe-prime' groups. This test must be performed for each 'safeprime' group that each protocol uses.

See Section 1.2 for a listing of applicable CAVP certificates

2.2.3 Cryptographic Key Destruction (MDMPP40:FCS_CKM_EXT.4)

2.2.3.1 MDMPP40:FCS_CKM_EXT.4.1

TSS Assurance Activities: If 'invoking platform-provided functionality' is selected, the evaluator shall examine the TSS to verify that it describes (for each supported platform) how the key destruction 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).

Section 6.2 of ST explains the EMM clears keys (TLS and HTTPS session keys) from memory after those keys are no longer needed. Furthermore, the EMM stores its certificates (the only persistently stored keying material) on an internal hard drive in encrypted format, and when an administrator configures new certificates, the EMM will directly overwrite the old keys with the new.

The EMM Agent relies upon its platform to securely clear keys (TLS and HTTPS session keys) from memory when no longer needed as the EMM Agent utilizes platform provided TLS and key storage.

Guidance Assurance Activities: None Defined

Testing Assurance Activities: None Defined

2.2.3.2 MDMPP40:FCS_CKM_EXT.4.2

TSS Assurance Activities: The evaluation activity used is dependent on the selection made in FCS_CKM_EXT.4.1. FCS_COP.1.1(1)

The evaluator shall check to ensure the TSS lists each type of plaintext key material and CSP (authentication data, authorization data, secret/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 'invoke platform-provided functionality' 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 'implement functionality' 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.

Section 6.2 of ST identifies CSP in the EMM. These include TLS and HTTPS session keys, as well as various certificates. Section 6.2 also indicates that the session keys are stored in memory and cleared when the session ends. Section 6.2 indicates that certificates are stored on disk and overwritten when new certificates replace the old certificate.

Guidance Assurance Activities: None Defined

Testing Assurance Activities: 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 1: The evaluator shall utilize 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 theTOE 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 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 utilize 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.

The evaluator created a custom SSL library which generated an output file for each TLS session key used for communications with a network peer. Using a client configured to use this custom library, the evaluator established an administrative web session with the TOE (i.e., an HTTPS connection to the TOE TLS server). The evaluator captured the traffic for this session, recorded the TLS session key provided by the custom library, and closed the connection. Immediately following the termination of the session, the evaluator obtained a memory dump of the virtual machine within which the TOE was executing. The evaluator searched the memory dump for the TLS session key and determined that the TLS session keys were not in memory.

Component TSS Assurance Activities: None Defined

Component Guidance Assurance Activities: None Defined

Component Testing Assurance Activities: None Defined

2.2.4 Cryptographic Operation (Confidentiality Algorithms) (MDMPP40:FCS_COP.1(1))

2.2.4.1 MDMPP40:FCS_COP.1.1(1)

TSS Assurance Activities: None Defined

Guidance Assurance Activities: None Defined

Testing Assurance Activities: None Defined

Component TSS Assurance Activities: 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/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).

Section 6.2 of the ST explains the EMM Server uses its RSA Crypto-J library and the EMM Agent relies upon its evaluated platform. The ST explains the Agent exclusively calls the evaluated Android APIs provided by the underlying phone platform.

Component Guidance Assurance Activities: None Defined

Component Testing Assurance Activities: FIPS

See Section 1.2 for a listing of applicable CAVP certificates.

2.2.5 Cryptographic Operation (Hashing Algorithms) (MDMPP40:FCS_COP.1(2))

2.2.5.1 MDMPP40:FCS_COP.1.1(2)

TSS Assurance Activities: None Defined

Guidance Assurance Activities: None Defined

Testing Assurance Activities: None Defined

Component TSS Assurance Activities: 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. the byte-oriented testmacs.

Section 6.2 of the ST explains the EMM Server uses its RSA Crypto-J library and the EMM Agent relies upon its evaluated platform. The ST explains the Agent exclusively calls the evaluated Android APIs provided by the underlying phone platform. Both the Agent and Server utilize these cryptographic algorithms primarily during establishment of TLS/HTTPS connections (which requires signature generation and verification as part of peer authentication, hashing as part of the signatures for peer authentication and for HMAC integrity, HMAC for integrity of the trusted channel, AES for the confidentiality of the trusted channel, and RBGs to generate nonces and IVs). The EMM also uses signature verification to ensure the authenticity of EMM software updates

Component Guidance Assurance Activities: None Defined

Component Testing Assurance Activities: FIPS

See Section 1.2 for a listing of applicable CAVP certificates.

2.2.6 Cryptographic Operation (Signature Algorithms) (MDMPP40:FCS_COP.1(3))

2.2.6.1 MDMPP40:FCS_COP.1.1(3)

TSS Assurance Activities: None Defined

Guidance Assurance Activities: None Defined

Testing Assurance Activities: None Defined

Component TSS Assurance Activities: 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).

Section 6.2 of the ST explains the EMM Server uses its RSA Crypto-J library and the EMM Agent relies upon its evaluated platform. The ST explains the Agent exclusively calls the evaluated Android APIs provided by the underlying phone platform.

Component Guidance Assurance Activities: None Defined

Component Testing Assurance Activities: FIPS

See Section 1.2 for a listing of applicable CAVP certificates.

2.2.7 Cryptographic Operation (Keyed-Hash Message Authentication) (MDMPP40:FCS_COP.1(4))

2.2.7.1 MDMPP40:FCS_COP.1.1(4)

TSS Assurance Activities: None Defined

Guidance Assurance Activities: None Defined

Testing Assurance Activities: None Defined

Component TSS Assurance Activities: 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.

Section 6.2 of the ST explains the EMM Server uses its RSA Crypto-J library and the EMM Agent relies upon its evaluated platform. When using HMAC as part of TLS, both the Agent and Server utilize HMAC keys equal to the block size of the underlying hash algorithm. When employing HMAC-SHA-256 or HMAC-SHA-384, the TOE uses a 32 or 48-byte key and block size of 64 or 128 bytes to produce a 32 or 48-byte hash, respectively.

Component Guidance Assurance Activities: None Defined

Component Testing Assurance Activities: FIPS

See Section 1.2 for a listing of applicable CAVP certificates.

2.2.8 HTTPS Protocol (MDMPP40:FCS_HTTPS_EXT.1)

2.2.8.1 MDMPP40:FCS_HTTPS_EXT.1.1

TSS Assurance Activities: None Defined

Guidance Assurance Activities: None Defined

Testing Assurance Activities: None Defined

2.2.8.2 MDMPP40:FCS_HTTPS_EXT.1.2

TSS Assurance Activities: None Defined

Guidance Assurance Activities: None Defined

Testing Assurance Activities: None Defined

Component TSS Assurance Activities: None Defined

Component Guidance Assurance Activities: None Defined

Component Testing Assurance Activities: Test 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.

The EMM Server acts as a TLS server for remote administration. The packet captures from test case FTP_TRP.1(1)-test 1 were analyzed and showed that traffic between a remote administrator and the EMM Server is protected by HTTPS, using TLS version 1.2. The packet captures also show that the data is not plaintext.

2.2.9 Initialization Vector Generation (MDMPP40:FCS_IV_EXT.1)

2.2.9.1 MDMPP40:FCS_IV_EXT.1.1

TSS Assurance Activities: None Defined

Guidance Assurance Activities: None Defined

Testing Assurance Activities: None Defined

Component TSS Assurance Activities: 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.

Section 6.2 of the ST states the EMM generates IVs for AES CBC using unpredictable (random) IVs drawn from the SHA-256 HMAC_DRBG (which meets the “unpredictable” requirement of SP 800-38A. The EMM Agent is entirely dependent on the underlying platform to generate IVs as necessary in support of cryptographic functions used by the EMM Agent.

Component Guidance Assurance Activities: None Defined

Component Testing Assurance Activities: The evaluator shall ensure that the generation of IVs for each key encrypted by the same KEK meets Table 4.

By reviewing the Security Target, the evaluator concluded that IVs are used as part of AES-CBC and AES-GCM encryption. These IVs are drawn from the SHA-256 HMAC_DRBG provided by RSA BSAFE Crypto-J. This DRBG meeting FCS_RBG_EXT.1. The unpredictability provided by use of these DRBG satisfies SP 800-38A and 800-38D.

2.2.10 Extended: Random Bit Generation (MDMPP40:FCS_RBG_EXT.1)

2.2.10.1 MDMPP40:FCS_RBG_EXT.1.1

TSS Assurance Activities: 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).

Section 6.2 of the ST states the EMM’s RSA BSAFE Crypto-J Cryptographic library provides a SHA-256 HMAC_DRBG seeded by the underlying platform (Microsoft Windows Server). Specifically, the Server’s cryptographic module seeds its DRBG using the Windows BCryptGenRandom() function. Because we cannot test Microsoft’s entropy implementation, we make an assumption of entropy regarding it (as required for any untestable third-party source) and assume that the output of BCryptGenRandom() contains at least 0.666 bits of entropy per bit of output With at least 0.666 bits of entropy per bit of output, the Server appropriately seeds its DRBG with at least 256-bits of entropy.

The EMM Agent makes indirect use of the AES-256 CTR_DRBG belonging to its underlying platform for all random bit generation (indirectly using it when calling platform provided cryptographic APIs).

Guidance Assurance Activities: None Defined

Testing Assurance Activities: FIPS

See Section 1.2 for a listing of applicable CAVP certificates.

2.2.10.2 MDMPP40:FCS_RBG_EXT.1.2

TSS Assurance Activities: Documentation shall be produced-and the evaluator shall perform the activities-in accordance with Appendix D: Entropy Documentation and Assessment and the 'Clarification to the Entropy Documentation and Assessment Annex.'

In the future, specific statistical testing (in line with NIST SP 800-90B) will be required to verify the entropy estimates.

The Entropy description is provided in a separate (non-ST) document that has been delivered to NIAP for approval. Note that the entropy analysis has been accepted by NIAP/NS.A

Guidance Assurance Activities: None Defined

Testing Assurance Activities: None Defined

Component TSS Assurance Activities: None Defined

Component Guidance Assurance Activities: None Defined

Component Testing Assurance Activities: None Defined

2.2.11 Cryptographic Key Storage (MDMPP40:FCS_STG_EXT.1)

2.2.11.1 MDMPP40:FCS_STG_EXT.1.1

TSS Assurance Activities: None Defined

Guidance Assurance Activities: None Defined

Testing Assurance Activities: None Defined

Component TSS Assurance Activities: 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.

Section 6.2 of the ST states the EMM encrypts its persistent keys (which consist exclusively of TLS/HTTPS certificates) by storing them encrypted with an AES-256 CBC key derived from a server secret. At no time does the Server store any plaintext keys on its hard drive (the only persistent memory the Server has). The Server does not store any ephemeral keys (e.g., TLS/HTTPS session keys). Each of the Agent's APKs store their keys (i.e., private keys associated with certificates) in the platform provided key storage (Android KeyStore) and then utilize those keys securely through the platform provided key storage.

Component Guidance Assurance Activities: None Defined

Component Testing Assurance Activities: None Defined

2.2.12 Cryptographic Key Storage (MDMA10:FCS_STG_EXT.1(2))

2.2.12.1 MDMA10:FCS_STG_EXT.1.1(2)

TSS Assurance Activities: None Defined

Guidance Assurance Activities: None Defined

Testing Assurance Activities: None Defined

Component TSS Assurance Activities: The evaluator will verify that the TSS 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, for each platform listed as supported in the ST, how it is stored. The evaluator shall verify that the Agent calls a platformprovided API to store persistent secrets and private keys.

Section 6.2 of the ST states each of the Agent's APKs store their keys (i.e., private keys associated with certificates) in the platform provided key storage (Android KeyStore) and then utilize those keys securely through the platform provided key storage.

Component Guidance Assurance Activities: None Defined

Component Testing Assurance Activities: None Defined

2.2.13 Encrypted Cryptographic Key Storage (MDMPP40:FCS_STG_EXT.2)

2.2.13.1 MDMPP40:FCS_STG_EXT.2.1

TSS Assurance Activities: None Defined

Guidance Assurance Activities: None Defined

Testing Assurance Activities: None Defined

Component TSS Assurance Activities: 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).

Section 6.2 of ST indicates that the only persistent secret keys that are stored by the EMM are certificates used for TLS/HTTPS. The secret keys for certificates are stored encrypted with an AES-CBC 256-bit key derived from a server secret. Section 6.2 also indicates that at no time does the EMM store any plaintext keys in persistent storage, such keys reside only in memory.

Component Guidance Assurance Activities: None Defined

Component Testing Assurance Activities: None Defined

2.2.14 TLS Protocol (PKGTLS11:FCS_TLS_EXT.1)

2.2.14.1 PKGTLS11:FCS_TLS_EXT.1.1

TSS Assurance Activities: None Defined

Guidance Assurance Activities: The evaluator shall ensure that the selections indicated in the ST are consistent with selections in the dependent components.

The evaluator reviewed the selections in the selections of PKGTLS11 requirements and the ST includes all necessary components.

Testing Assurance Activities: None Defined

Component TSS Assurance Activities: None Defined

Component Guidance Assurance Activities: None Defined

Component Testing Assurance Activities: None Defined

2.2.15 TLS Client Protocol (PKGTLS11:FCS_TLSC_EXT.1)

2.2.15.1 PKGTLS11:FCS_TLSC_EXT.1.1

TSS Assurance Activities: The evaluator shall check the description of the implementation of this protocol in the TSS to ensure that the cipher suites supported are specified. The evaluator shall check the TSS to ensure that the cipher suites specified include those listed for this component.

Section 6.2 of ST claims that the TLS implementation used by the EMM supports all of the TLS cipher suites identified by the PKGTLS11:FCS_TLSC_EXT.1 requirement in Section 5.

Guidance Assurance Activities: The evaluator shall also check the operational guidance to ensure that it contains instructions on configuring the product so that TLS conforms to the description in the TSS.

Appendix B, "Configuring allowable Cipher" of [Install] provides instructions to configure the TLS connections used by the EMM. This includes the ciphersuites that are supported in Common Criteria mode.

Testing Assurance Activities: The evaluator shall also perform the following tests:

Test 1: The evaluator shall establish a TLS connection using each of the cipher suites specified by the requirement. This connection may be established as part of the establishment of a higher-level protocol, e.g., as part of an EAP session. It is sufficient to observe the successful negotiation of a cipher suite to satisfy the intent of the test; it is not necessary to examine the characteristics of the encrypted traffic in an attempt to discern the cipher suite being used (for example, that the cryptographic algorithm is 128-bit AES and not 256-bit AES).

Test 2: The goal of the following test is to verify that the TOE accepts only certificates with appropriate values in the extendedKeyUsage extension, and implicitly that the TOE correctly parses the extendedKeyUsage extension as part of X.509v3 server certificate validation. The evaluator shall attempt to establish the connection using a server with a server certificate that contains the Server Authentication purpose in the extendedKeyUsage extension and verify that a connection is established. The evaluator shall repeat this test using a different, but otherwise valid and trusted, certificate that lacks the Server Authentication purpose in the extendedKeyUsage extension and ensure that a connection is not established. Ideally, the two certificates should be similar in structure, the types of identifiers used, and the chain of trust.

Test 3: The evaluator shall send a server certificate in the TLS connection that does not match the server-selected cipher suite (for example, send a ECDSA certificate while using the TLS_RSA_WITH_AES_128_CBC_SHA cipher suite or send a RSA certificate while using one of the ECDSA cipher suites.) The evaluator shall verify that the product disconnects after receiving the server's Certificate handshake message.

Test 4: The evaluator shall configure the server to select the TLS_NULL_WITH_NULL_NULL cipher suite and verify that the client denies the connection.

Test 5: The evaluator shall perform the following modifications to the traffic:

Test 5.1: Change the TLS version selected by the server in the Server Hello to an undefined TLS version (for example 1.5 represented by the two bytes 03 06) and verify that the client rejects the connection.

Test 5.2: Change the TLS version selected by the server in the Server Hello to the most recent unsupported TLS version (for example 1.1 represented by the two bytes 03 02) and verify that the client rejects the connection.

Test 5.3: [conditional] If DHE or ECDHE cipher suites are supported, modify at least one byte in the server's nonce in the Server Hello handshake message, and verify that the client does not complete the handshake and no application data flows.

Test 5.4: Modify the server's selected cipher suite in the Server Hello handshake message to be a cipher suite not presented in the Client Hello handshake message. The evaluator shall verify that the client does not complete the handshake and no application data flows.

Test 5.5: [conditional] If DHE or ECDHE cipher suites are supported, modify the signature block in the server's Key Exchange handshake message, and verify that the client does not complete the handshake and no application data flows. This test does not apply to cipher suites using RSA key exchange. If a TOE only supports RSA key exchange in conjunction with TLS, then this test shall be omitted.

Test 5.6: Modify a byte in the Server Finished handshake message, and verify that the client does not complete the handshake and no application data flows.

Test 5.7: Send a message consisting of random bytes from the server after the server has issued the Change Cipher Spec message and verify that the client does not complete the handshake and no application data flows. The message must still have a valid 5-byte record header in order to ensure the message will be parsed as TLS.

The evaluator performed the following tests against the EMM's TLS client interface (including the AppTunnel and Push TLS client implementations used in the TOE components).

Test 1: The evaluator established a TLS session on each of the AppTunnel and Push TLS client implementations for each of the claimed ciphersuites in turn. The evaluator used a network sniffer to capture the TLS session negotiation and observed that the expected TLS cipher is negotiated.

Test 2: The evaluator configured the server to require mutual authentication and established a TLS session with each of the AppTunnel and Push TLS client implementations. The evaluator configured the remote peer (i.e., the client) to send a certificate with the Server Authentication purpose in the extendedKeyUsage field. Using a network sniffer the evaluator captured the TLS session negotiation and observed that the TLS session is accepted by the server. The evaluator reconfigured the test server to retry the TLS session using a client cert that is missing the Server Authentication purpose in the extendedKeyUsage field. Using a network sniffer the evaluator captured the TLS session negotiation and observed that the TLS session is rejected by the server.

Test 3: The evaluator established a TLS session from each of the AppTunnel and Push TLS client implementations. A modified test server negotiates an RSA ciphersuite, but returns an ECDSA Certificate. Using a network sniffer to capture the TLS session negotiation and observed that the TLS session is not negotiated successfully.

Test 4: The evaluator configured a test server to accept only the TLS_NULL_WITH_NUL_NULL ciphersuite. The evaluator then attempted to establish a TLS session from each of the AppTunnel and Push TLS client implementations to that test server. Using a network sniffer the evaluator captured the TLS session negotiation and observed that the TLS session is rejected.

Test 5: The evaluator obtained a packet captures of the TLS session negotiation between teach of the AppTunnel and Push TLS client implementations and a test server with Mutual Authentication configured on the test server. The evaluator made connection attempts from the client to the test server. The server implementation of the TLS protocol was modified as stated in the 7 scenarios described by the Assurance Activity. The evaluator inspected each packet captures to ensure that the connections are rejected for each scenario.

2.2.15.2 PKGTLS11:FCS_TLSC_EXT.1.2

TSS Assurance Activities: The evaluator shall ensure that the TSS describes the client's method of establishing all reference identifiers from the application-configured reference identifier, including which types of reference identifiers are supported (e.g. Common Name, DNS Name, URI Name, Service Name, or other application-specific Subject Alternative Names) and whether IP addresses and wildcards are supported. The evaluator shall ensure that

this description identifies whether and the manner in which certificate pinning is supported or used by the product.

Section 6.2 of ST explains that the EMM performs hostname checking to ensure that either the expected hostname matches the Distinguished Name (DN) in the presented certificate. When acting as a TLS client, the EMM server components support the use of wildcards in accordance with RFC 6125. Section 6.2 also indicates that the EMM does not utilize certificate pinning,

Guidance Assurance Activities: The evaluator shall verify that the AGD guidance includes instructions for setting the reference identifier to be used for the purposes of certificate validation in TLS.

Section 2.2 of [Install] describes the requirements for certificates used by the EMM to identify themselves to other parts of the TOE and to entities in the environment. Within Section 2.2.1 of [Install], "Verifying certificate distinguished name (DN)" includes information describing the configuration constraints for the EMM regarding these component's identification using certificate values. This includes constraints enforced on the reference identifier used by the TOE components.

Testing Assurance Activities: The evaluator shall configure the reference identifier according to the AGD guidance and perform the following tests during a TLS connection:

Test 1: The evaluator shall present a server certificate that contains a CN that does not match the reference identifier and does not contain the SAN extension. The evaluator shall verify that the connection fails. Note that some systems might require the presence of the SAN extension. In this case the connection would still fail but for the reason of the missing

Testing Assurance Activities: The evaluator shall configure the reference identifier according to the AGD guidance and perform the following tests during a TLS connection. If the TOE supports certificate pinning, all pinned certificates must be removed before performing Tests 1 through 6. A pinned certificate must be added prior to performing Test 7. (TD0499 applied)

Test 1: The evaluator shall present a server certificate that contains a CN that does not match the reference identifier and does not contain the SAN extension. The evaluator shall verify that the connection fails. Note that some systems might require the presence of the SAN extension. In this case the connection would still fail but for the reason of the missing SAN extension instead of the mismatch of CN and reference identifier. Both reasons are acceptable to pass Test 1.

Test 2: The evaluator shall present a server certificate that contains a CN that matches the reference identifier, contains the SAN extension, but does not contain an identifier in the SAN that matches the reference identifier. The evaluator shall verify that the connection fails. The evaluator shall repeat this test for each supported SAN type.

Test 3: [conditional] If the TOE does not mandate the presence of the SAN extension, the evaluator shall present a server certificate that contains a CN that matches the reference identifier and does not contain the SAN extension.

The evaluator shall verify that the connection succeeds. If the TOE does mandate the presence of the SAN extension, this Test shall be omitted.

Test 4: The evaluator shall present a server certificate that contains a CN that does not match the reference identifier but does contain an identifier in the SAN that matches. The evaluator shall verify that the connection succeeds.

Test 5: The evaluator shall perform the following wildcard tests with each supported type of reference identifier. The support for wildcards is intended to be optional. If wildcards are supported, the first, second, and third tests below shall be executed. If wildcards are not supported, then the fourth test below shall be executed.

Test 5.1: [conditional]: If wildcards are supported, the evaluator shall present a server certificate containing a wildcard that is not in the left-most label of the presented identifier (e.g. foo.*.example.com) and verify that the connection fails.

Test 5.2: [conditional]: If wildcards are supported, the evaluator shall present a server certificate containing a wildcard in the left-most label but not preceding the public suffix (e.g. *.example.com). The evaluator shall configure the reference identifier with a single left-most label (e.g. foo.example.com) and verify that the connection succeeds. The evaluator shall configure the reference identifier without a left-most label as in the certificate (e.g. example.com) and verify that the connection fails. The evaluator shall configure the reference identifier with two left-most labels (e.g. bar.foo.example.come) and verify that the connection fails.

Test 5.3: [conditional]: If wildcards are supported, the evaluator shall present a server certificate containing a wildcard in the left-most label immediately preceding the public suffix (e.g. *.com). The evaluator shall configure the reference identifier with a single left-most label (e.g. foo.com) and verify that the connection fails. The evaluator shall configure the reference identifier with two left-most labels (e.g. bar.foo.com) and verify that the connection fails.

Test 5.4: [conditional]: If wildcards are not supported, the evaluator shall present a server certificate containing a wildcard in the left-most label (e.g. *.example.com). The evaluator shall configure the reference identifier with a single left-most label (e.g. foo.example.com) and verify that the connection fails.

Test 6: [conditional] If URI or Service name reference identifiers are supported, the evaluator shall configure the DNS name and the service identifier. The evaluator shall present a server certificate containing the correct DNS name and service identifier in the URIName or SRVName fields of the SAN and verify that the connection succeeds. The evaluator shall repeat this test with the wrong service identifier (but correct DNS name) and verify that the connection fails.

Test 7: [conditional] If pinned certificates are supported the evaluator shall present a certificate that does not match the pinned certificate and verify that the connection fails.

The evaluator performed the following tests against the EMM's TLS client interface (including the AppTunnel and Push TLS client implementations used in the TOE components).

Test 1: The evaluator established a TSL session from each of the AppTunnel and Push TLS client implementations targeting a server using a valid certificate with a CN matching the domain name used by the client. Using a network sniffer to capture the TLS session negotiation the evaluator examined the traffic capture and observed a successful connection. The evaluator then established a TLS session from the EMM component targeting a server using a server certificate that does not contain an identifier in either the Subject Alternative Name (SAN) or Common Name (CN) that matches the reference identifier. Using a network sniffer to capture the TLS session negotiation the evaluator examined the traffic capture and observed that the TLS session was not negotiated successfully.

Test 2: The evaluator established a TLS session from each of the AppTunnel and Push TLS client implementations targeting a server using a server certificate that contains a CN that matches the reference identifier, contains the SAN extension, but does not contain an identifier in the SAN that matches the reference identifier. Using a network sniffer to capture the TLS session negotiation the evaluator examined the traffic capture and observed that the TLS session was not negotiated successfully.

Test 3: The evaluator established a TLS session from each of the AppTunnel and Push TLS client implementations targeting a server using a server certificate that contains a CN that matches the reference identifier and does not contain the SAN extension. Using a network sniffer to capture the TLS session negotiation the evaluator examined the traffic capture and observed that the TLS session was negotiated successfully.

Test 4: The evaluator established a TLS session from each of the AppTunnel and Push TLS client implementations targeting a server using a server certificate that contains a CN that does not match the reference identifier but does contain an identifier in the SAN that matches. Using a network sniffer to capture the TLS session negotiation the evaluator examined the traffic capture and observed that the TLS session was negotiated successfully.

Test 5: The evaluator configured a test server to use a server certificate containing a reference identifier as described by test 5 from the above assurance activity. The evaluator established a TLS session from each of the AppTunnel and Push TLS client implementations targeting the name shown in column 2 of the following table. Using a network sniffer to capture the TLS session negotiation the evaluator examined the traffic capture and observed that the TLS session was negotiated as shown in column 3 of the following table.

Certificate Contents	Host ID	Expected Result
CN=bar.*.example.com	bar.foo.example.com	No Connection
SAN=bar.*.example.com	bar.foo.example.com	No Connection
CN=*.example.com	foo.example.com	Successful Connection
SAN=*.example.com	foo.example.com	Successful Connection
CN=*.com	example.com	No Connection
SAN=*.com	example.com	No Connection
CN=*.example.com	bar.foo.example.com	No Connection
SAN=*.example.com	bar.foo.example.com	No Connection
CN=*.com	example.com	No Connection
SAN=*.com	example.com	No Connection
CN=*.example.com	foo.example.com	No Connection
SAN=*.example.com	foo.example.com	No Connection

Test 6: The TOE does not support the optional URI or Service Name used as reference identifiers.

Test 7: The TOE does not support certificate pinning

2.2.15.3 PKGTLS11:FCS_TLSC_EXT.1.3

TSS Assurance Activities: If the selection for authorizing override of invalid certificates is made, then the evaluator shall ensure that the TSS includes a description of how and when user or administrator authorization is obtained. The evaluator shall also ensure that the TSS describes any mechanism for storing such authorizations, such that future presentation of such otherwise-invalid certificates permits establishment of a trusted channel without user or administrator action.

The product does not offer a mechanism to override invalid certificates.

Guidance Assurance Activities: None Defined

Testing Assurance Activities: The evaluator shall demonstrate that using an invalid certificate results in the function failing as follows, unless excepted:

Test 1a: The evaluator shall demonstrate that a server using a certificate with a valid certification path successfully connects.

Test 1b: The evaluator shall modify the certificate chain used by the server in test 1a to be invalid and demonstrate that a server using a certificate without a valid certification path to a trust store element of the TOE results in an authentication failure.

Test 1c [conditional]: If the TOE trust store can be managed, the evaluator shall modify the trust store element used in Test 1a to be untrusted and demonstrate that a connection attempt from the same server used in Test 1a results in an authentication failure.

(TD0513 applied)

Test 2: The evaluator shall demonstrate that a server using a certificate which has been revoked results in an authentication failure.

Test 3: The evaluator shall demonstrate that a server using a certificate which has passed its expiration date results in an authentication failure.

Test 4: The evaluator shall demonstrate that a server using a certificate which does not have a valid identifier results in an authentication failure.

Test 1: The evaluator configured each of the AppTunnel and Push TLS client implementations to connect to a test server and established a TLS session then configured the remote peer (i.e., the server) to use a certificate with a

broken certificate path. The evaluator used a network sniffer to capture the TLS session negotiation and observed that the TLS session is rejected by the server.

Test 2: The evaluator configured each of the AppTunnel and Push TLS client implementations to connect to a test server and established a TLS session then configured the remote peer (i.e., the server) to use a certificate during the TLS negotiation where the certificate was valid. A packet capture was obtained of this TLS negotiation which shows that the connection was successful. The evaluator then had the server provide a revoked certificate and attempted the same connection from the test client. The connection attempt performed after revoking the certificate was not successful.

Test 3: The evaluator configured each of the AppTunnel and Push TLS client implementations to connect to a test server and established a TLS session then configured the remote peer (i.e., the server) to use an expired client certificate. The evaluator used a network sniffer to capture the TLS session negotiations and observed the connection was not successful.

Test 4: The evaluator configured each of the AppTunnel and Push TLS client implementations to connect to a test server and established a TLS session then configured the remote peer (i.e., the server) to use a certificate that does not have a valid identifier. The evaluator used a network sniffer to capture the TLS session negotiations and observed the connection was not successful.

Component TSS Assurance Activities: None Defined

Component Guidance Assurance Activities: None Defined

Component Testing Assurance Activities: None Defined

2.2.16 TLS Client Support for Mutual Authentication (PKGTLS11:FCS_TLSC_EXT.2)

2.2.16.1 PKGTLS11:FCS_TLSC_EXT.2.1

TSS Assurance Activities: The evaluator shall ensure that the TSS description required per FIA_X509_EXT.2.1 includes the use of client-side certificates for TLS mutual authentication. The evaluator shall also ensure that the TSS describes any factors beyond configuration that are necessary in order for the client to engage in mutual authentication using X.509v3 certificates.

Section 6.2 of the ST states the EMM web UI interface does not support mutual authentication for remote administrator access nor for initial device enrollment, so in those cases there is no certificate checking performed. After initial enrollment Agents communicate exclusively with the AT and PUSH server component interfaces requiring mutual authentication using X509 certificates. The EMM performs hostname checking to ensure that the expected hostname matches the Distinguished Name (DN) in the presented certificate. When performing

revocation checking, the TOE checks the peer's certificate against a CRL to determine if the certificate remains valid.

Guidance Assurance Activities: The evaluator shall ensure that the AGD guidance includes any instructions necessary to configure the TOE to perform mutual authentication. The evaluator also shall verify that the AGD guidance required per FIA_X509_EXT.2.1 includes instructions for configuring the client-side certificates for TLS mutual authentication.

TLS mutual authentication using X.509v3 certificates is always performed by the EMM once it is installed according to instructions [Install] section 3. No further configuration is necessary.

Testing Assurance Activities: The evaluator shall also perform the following tests:

Test 1: The evaluator shall establish a connection to a server that is not configured for mutual authentication (i.e. does not send Server's Certificate Request (type 13) message). The evaluator observes negotiation of a TLS channel and confirms that the TOE did not send Client's Certificate message (type 11) during handshake.

Test 2: The evaluator shall establish a connection to a server with a shared trusted root that is configured for mutual authentication (i.e. it sends Server's Certificate Request (type 13) message). The evaluator observes negotiation of a TLS channel and confirms that the TOE responds with a non-empty Client's Certificate message (type 11) and Certificate Verify (type 15) message.

Test 1: The evaluator established a TLS connection between each of the AppTunnel and Push TLS client implementations and a test server that was not configured for mutual authentication. The evaluator observed that the TLS connection was successful and the TOE did not send a certificate or a certificate verify message.

Test 2: The evaluator established a TLS connection between each of the AppTunnel and Push TLS client implementations and a test server that was configured for mutual authentication. The evaluator observed that the TLS connection was successful and the TOE did send both a certificate and a certificate verify message/

Component TSS Assurance Activities: None Defined

Component Guidance Assurance Activities: None Defined

Component Testing Assurance Activities: None Defined

2.2.17 TLS Client Support for Supported Groups Extension (PKGTLS11:FCS_TLSC_EXT.5)

2.2.17.1 PKGTLS11:FCS_TLSC_EXT.5.1

TSS Assurance Activities: The evaluator shall verify that TSS describes the Supported Groups Extension.

Section 6.2 of the ST states the EMM supports only NIST curves secp256r1, and secp384r1 when using elliptic curve ciphers. The EMM supports RSA or DHE key exchange using certificates of either 2048, 3072 or 4096 bits.

Guidance Assurance Activities: None Defined

Testing Assurance Activities: The evaluator shall also perform the following test:

Test 1: The evaluator shall configure a server to perform key exchange using each of the TOE's supported curves and/or groups. The evaluator shall verify that the TOE successfully connects to the server.

The evaluator configured a server to use various certificates necessary to cause a TLS negotiation using each of the key exchange methods. The evaluator then initiated a TLS session from each of the AppTunnel and Push TLS client implementations while capturing traffic. Inspection of the traffic indicated that the TOE did in fact negotiate a successful connection using each of the claimed key exchange methods.

Component TSS Assurance Activities: None Defined

Component Guidance Assurance Activities: None Defined

Component Testing Assurance Activities: None Defined

2.2.18 TLS Server Protocol - per TD0726 (PKGTLS11:FCS_TLSS_EXT.1)

2.2.18.1 PKGTLS11:FCS_TLSS_EXT.1.1

TSS Assurance Activities: The evaluator shall check the description of the implementation of this protocol in the TSS to ensure that the cipher suites supported are specified. The evaluator shall check the TSS to ensure that the cipher suites specified include those listed for this component.

Section 6.2 of ST claims that the TLS implementation used by the EMM (including the EMM Server, AppTunnel, and Push TLS server implementations used in the TOE components) support all of the TLS ciphersuites identified by the PKGTLS:FCS_TLSS_EXT.1 requirement. Section 6.2 also indicates that the EMM supports only TLS version 1.2.

Guidance Assurance Activities: The evaluator shall also check the operational guidance to ensure that it contains instructions on configuring the TOE so that TLS conforms to the description in the TSS

Appendix B.1 of [Install], describes how to configure the allowable ciphers used by the App Tunnel and Push server components of the EMM. The TLS protocol version and ciphers are specified by setting values within configuration

files. The guidance describes how to configure the TLS protocol version or ciphersuites used by the EMM Server component of the EMM in Appendix B.2 of [Install].

Testing Assurance Activities: The evaluator shall also perform the following tests:

Test 1: The evaluator shall establish a TLS connection using each of the cipher suites specified by the requirement. This connection may be established as part of the establishment of a higher-level protocol, e.g., as part of an EAP session. It is sufficient to observe the successful negotiation of a cipher suite to satisfy the intent of the test; it is not necessary to examine the characteristics of the encrypted traffic in an attempt to discern the cipher suite being used (for example, that the cryptographic algorithm is 128-bit AES and not 256-bit AES).

Test 2: The evaluator shall send a Client Hello to the server with a list of cipher suites that does not contain any of the cipher suites in the server's ST and verify that the server denies the connection. Additionally, the evaluator shall send a Client Hello to the server containing only the TLS_NULL_WITH_NULL_NULL cipher suite and verify that the server denies the connection.

Test 3: If RSA key exchange is used in one of the selected ciphersuites, the evaluator shall use a client to send a properly constructed Key Exchange message with a modified EncryptedPreMasterSecret field during the TLS handshake. The evaluator shall verify that the handshake is not completed successfully and no application data flows.

Test 4: The evaluator shall perform the following modifications to the traffic:

Test 4.1: Removed per TD0469

Test 4.2: Modify a byte in the data of the client's Finished handshake message, and verify that the server rejects the connection and does not send any application data.

Test 4.3: Demonstrate that the TOE will not resume a session for which the client failed to complete the handshake (independent of TOE support for session resumption):

Test 4.3i [conditional]: If the TOE does not support session resumption based on session IDs according to RFC4346 (TLS1.1) or RFC5246 (TLS1.2) or session tickets according to RFC5077, the evaluator shall perform the following test:

a) The evaluator shall send a Client Hello with a zero-length session identifier and with a SessionTicket extension containing a zero-length ticket.

b) The evaluator shall verify the server does not send a NewSessionTicket handshake message (at any point in the handshake).

c) The evaluator shall verify the Server Hello message contains a zero-length session identifier or passes the following steps:

Note: The following steps are only performed if the ServerHello message contains a non-zero length SessionID.

d) The evaluator shall complete the TLS handshake and capture the SessionID from the ServerHello.

e) The evaluator shall send a ClientHello containing the SessionID captured in step d). This can be done by keeping the TLS session in step d) open or start a new TLS session using the SessionID captured in step d).

f) The evaluator shall verify the TOE (1) implicitly rejects the SessionID by sending a ServerHello containing a different SessionID and by performing a full handshake (as shown in Figure 1 of RFC 4346 or RFC 5246), or (2) terminates the connection in some way that prevents the flow of application data.

Test 4.3ii [conditional]: If the TOE supports session resumption using session IDs according to RFC4346 (TLS1.1) or RFC5246 (TLS1.2), the evaluator shall carry out the following steps (note that for each of these tests, it is not necessary to perform the test case for each supported version of TLS):

a) The evaluator shall conduct a successful handshake and capture the TOE-generated session ID in the Server Hello message. The evaluator shall then initiate a new TLS connection and send the previously captured session ID to show that the TOE resumed the previous session by responding with ServerHello containing the same SessionID immediately followed by ChangeCipherSpec and Finished messages (as shown in Figure 2 of RFC 4346 or RFC 5246).

b) The evaluator shall initiate a handshake and capture the TOE-generated session ID in the Server Hello message. The evaluator shall then, within the same handshake, generate or force an unencrypted fatal Alert message immediately before the client would otherwise send its ChangeCipherSpec message thereby disrupting the handshake. The evaluator shall then initiate a new Client Hello using the previously captured session ID, and verify that the server (1) implicitly rejects the session ID by sending a ServerHello containing a different SessionID and performing a full handshake (as shown in figure 1 of RFC 4346 or RFC 5246), or (2) terminates the connection in some way that prevents the flow of application data.

Test 4.3iii [conditional]: If the TOE supports session tickets according to RFC5077, the evaluator shall carry out the following steps (note that for each of these tests, it is not necessary to perform the test case for each supported version of TLS):

a) The evaluator shall permit a successful TLS handshake to occur in which a session ticket is exchanged with the non-TOE client. The evaluator shall then attempt to correctly reuse the previous session by sending the session ticket in the ClientHello. The evaluator shall confirm that the TOE responds with a ServerHello with an empty SessionTicket extension, NewSessionTicket, ChangeCipherSpec and Finished messages (as seen in figure 2 of RFC 5077).

b) The evaluator shall permit a successful TLS handshake to occur in which a session ticket is exchanged with the non-TOE client. The evaluator will then modify the session ticket and send it as part of a new Client Hello message. The evaluator shall confirm that the TOE either (1) implicitly rejects the session ticket by performing a full handshake (as shown in figure 3 or 4 of RFC 5077), or (2) terminates the connection in some way that prevents the flow of application data.

Test 4.4: Send a message consisting of random bytes from the client after the client has issued the ChangeCipherSpec message and verify that the server denies the connection.

(TD0588 applied)

Test 1: The evaluator established a TLS session on each of the EMM Server, AppTunnel, and Push TLS server implementations for each of the claimed ciphersuites in turn. The evaluator used a network sniffer to capture the TLS session negotiation and observed that the expected TLS cipher is negotiated.

Test 2: Establish a TLS session on each of the EMM Server, AppTunnel, and Push TLS server implementations for each accessible server interface and each secure client interface with the TLS_NULL_WITH_NULL_NULL ciphersuite. The evaluator used a network sniffer to capture the TLS session negotiation and observed that the session is not successfully negotiated. The evaluator established a TLS session with each of the EMM Server, AppTunnel, and Push TLS server implementations and configure the remote peer (i.e., the client) to change the ciphersuite to one that doesn't match the server-selected ciphersuite. The evaluator used a network sniffer to capture the TLS session negotiation and observed that the TLS session cannot be negotiated

Test 3: The evaluator established a TLS session with each of the EMM Server, AppTunnel, and Push TLS server implementations and configure the remote peer (i.e., the client) to use a modified encrypted premaster secret field in its otherwise correct client key exchange message. The evaluator used a network sniffer to capture the TLS session negotiation and observed that the TLS session cannot be negotiated.

Test 4: The evaluator obtained a packet capture of the TSL session negotiation between a test client and each of the EMM Server, AppTunnel, and Push TLS server implementations (individually) with Mutual Authentication configured on the MDM components.

4.1 - Removed per TD0469

4.2 - The evaluator attempted a connection to the TOE where the evaluator modified a byte in the Finished handshake message, verified that the TOE rejected the connection attempt after receiving the modified Finished message, and that the TOE sent no application data.

4.3i - The evaluator attempted to open a TLS connection to the TOE where the evaluator's client recorded the session_id from a failed TLS connection attempt. The evaluator observed the TOE reject the connection. The evaluator then attempted to open a TLS connection to the TOE where the evaluator's client reuses the session_id from the prior, failed TLS connection. The TOE accepted the connection, but rejected the attempt to reuse the bad session ID and returned a new session ID.

4.3ii - Not Applicable, as the TOE does not support session resumption using Session IDs.

4.3iii - Not Applicable, as the TOE does not support session tickets.

4.4 - The evaluator garbled a message between the TOE and its TLS peer. The evaluator observed that the Client denies the connection. Due to the nature of the error, regardless of whether the TOE is the client or server, the client is always the first to recognize the error.

2.2.18.2 PKGTLS11:FCS_TLSS_EXT.1.2

TSS Assurance Activities: The evaluator shall verify that the TSS contains a description of the denial of old SSL and TLS versions consistent relative to selections in FCS_TLSS_EXT.1.2.

Section 6.2 of the ST states the TOE supports TLS version 1.2. All versions of the SSL protocol and older versions of the TLS protocol are refused by the EMM.

Guidance Assurance Activities: The evaluator shall verify that the AGD guidance includes any configuration necessary to meet this requirement.

As explained in Section 6.2 of ST states that all versions of the SSL protocol and older versions of the TLS protocol (older than version 1.2) are refused by the EMM. Therefore, no instructions are necessary in the AGD guidance

Testing Assurance Activities: Test 1: The evaluator shall send a Client Hello requesting a connection with version SSL 2.0 and verify that the server denies the connection. The evaluator shall repeat this test with SSL 3.0 and TLS 1.0, and TLS 1.1 if it is selected.

The evaluator attempted to establish a TLS session on each of the EMM Server, AppTunnel, and Push TLS server implementations with each of the unsupported versions of SSL and TLS. The evaluator used a network sniffer to capture the session negotiation and observed that the expected protocol and version were offered and rejected during negotiation.

2.2.18.3 PKGTLS11:FCS_TLSS_EXT.1.3

TSS Assurance Activities: The evaluator shall verify that the TSS describes the key agreement parameters of the server's Key Exchange message

Section 6.2 of ST indicates that the EMM supports only NIST curves secp256r1 and secp384r1 and when using elliptic curve ciphers. The EMM supports signature_algorithms extensions in the client hello with SHA256 and SHA384 hashing. The EMM supports RSA or DHE key exchange using certificates of either 2048, 3072 or 4096 bits.

Guidance Assurance Activities: The evaluator shall verify that any configuration guidance necessary to meet the requirement must be contained in the AGD guidance.

Appendix B.1 of [Install] describes how to specify the ciphersuites supported for each of the EMM Server, AppTunnel, and Push TLS server implementations found in the EMM Server, Push server and AppTunnel server. This material indicates the default ciphersuites supported by the EMM server, Push server and AppTunnel server

for use as both servers and (where applicable) client communication with each other. The default ciphersuites do not require further configuration to include only ciphersuites meeting the FCS_TLSS_EXT.1 requirement.

The ciphersuites supported by each of the EMM Server, AppTunnel, and Push TLS server implementations are:

- TLS_RSA_WITH_AES_128_CBC_SHA256
- TLS_RSA_WITH_AES_256_CBC_SHA256
- TLS_RSA_WITH_AES_256_GCM_SHA384
- TLS_DHE_RSA_WITH_AES_128_CBC_SHA256
- TLS_DHE_RSA_WITH_AES_256_CBC_SHA256
- TLS_DHE_RSA_WITH_AES_256_GCM_SHA384
- TLS_ECDHE_ECDSA_WITH_AES_128_CBC_SHA256
- TLS_ECDHE_ECDSA_WITH_AES_128_GCM_SHA256
- TLS_ECDHE_ECDSA_WITH_AES_256_CBC_SHA384
- TLS_ECDHE_ECDSA_WITH_AES_256_GCM_SHA384
- TLS_ECDHE_RSA_WITH_AES_128_CBC_SHA256
- TLS_ECDHE_RSA_WITH_AES_128_GCM_SHA256
- TLS_ECDHE_RSA_WITH_AES_256_CBC_SHA384
- TLS_ECDHE_RSA_WITH_AES_256_GCM_SHA384

These ciphersuites are consistent with those identified in the PKGTLS11:FCS_TLSS_EXT.1 requirement

Testing Assurance Activities: The evaluator shall conduct the following tests. The testing can be carried out manually with a packet analyzer or with an automated framework that similarly captures such empirical evidence. Note that this testing can be accomplished in conjunction with other testing activities. For each of the following tests, determining that the size matches the expected size is sufficient.

Test 1: [conditional] If RSA-based key establishment is selected, the evaluator shall configure the TOE with a certificate containing a supported RSA size and attempt a connection. The evaluator shall verify that the size used matches that which is configured and that the connection is successfully established. The evaluator shall repeat this test for each supported size of RSA-based key establishment.

Test 2: [conditional] If finite-field (i.e. non-EC) Diffie-Hellman ciphers are selected, the evaluator shall attempt a connection using a Diffie-Hellman key exchange with a supported parameter size or supported group. The evaluator shall verify that the key agreement parameters in the Key Exchange message are the ones configured. The evaluator shall repeat this test for each supported parameter size or group.

Test 3: [conditional] If ECDHE ciphers are selected, the evaluator shall attempt a connection using an ECDHE ciphersuite with a supported curve. The evaluator shall verify that the key agreement parameters in the Key Exchange message are the ones configured. The evaluator shall repeat this test for each supported elliptic curve.

Test 1: The evaluator attempted a connection using an RSA-based key establishment with a supported size (2048, 3072, 4096) for each of the EMM Server, AppTunnel, and Push TLS server implementations. The evaluator shall verify that the size used matches that which is configured. The evaluator used a network sniffer to capture the TLS session negotiation and observed that the TLS session is rejected by the server. This was repeated for each of these Key exchange methods.

Test 2: The evaluator attempted a connection using a Diffie-Hellman key exchange with a supported parameter size or supported group for each of the EMM Server, AppTunnel, and Push TLS server implementations. The evaluator shall verify that the key agreement parameters in the Key Exchange message are the ones configured. The evaluator used a network sniffer to capture the TLS session negotiation and observed that the TLS session is rejected by the server. This was repeated for each of these Key exchange methods.

Test 3: The evaluator attempted a connection using an ECDHE ciphersuite with a supported curve (i.e., secp256r1 or secp384r1) for each of the EMM Server, AppTunnel, and Push TLS server implementations. Using a network sniffer to capture the TLS session negotiation and observed that the TLS session is rejected by the server. This was repeated for each of these Key exchange methods.

Component TSS Assurance Activities: None Defined

Component Guidance Assurance Activities: None Defined

Component Testing Assurance Activities: None Defined

2.2.19 TLS Server Support for Mutual Authentication (PKGTLS11:FCS_TLSS_EXT.2)

2.2.19.1 PKGTLS11:FCS_TLSS_EXT.2.1

TSS Assurance Activities: None Defined

Guidance Assurance Activities: None Defined

Testing Assurance Activities: None Defined

2.2.19.2 PKGTLS11:FCS_TLSS_EXT.2.2

TSS Assurance Activities: The evaluator shall ensure that the TSS description required per FIA_X509_EXT.2.1 includes the use of client-side certificates for TLS mutual authentication.

Section 6.3 of ST explains that the EMM (except for the EMM Server web UI and in initial enrollment connections) uses X.509 certificates only during TLS/HTTPS trusted channel establishment for server and client/mutual authentication.

Guidance Assurance Activities: The evaluator shall verify that the AGD guidance required per FIA_X509_EXT.2.1 includes instructions for configuring the client-side certificates for TLS mutual authentication. The evaluator shall ensure that the AGD guidance includes instructions for configuring the server to require mutual authentication of clients using these certificates.

TLS mutual authentication using X.509v3 certificates is always performed by the EMM (except for the EMM Server web UI and in initial enrollment connections) once it is installed according to instructions [Install] section 3. No further configuration is necessary. Note that the same certificates are used for both server and client connections by the EMM server components.

Testing Assurance Activities: The evaluator shall use TLS as a function to verify that the validation rules in FIA_X509_EXT.1.1 are adhered to and shall perform the following tests. The evaluator shall apply the AGD guidance to configure the server to require TLS mutual authentication of clients for the following tests, unless overridden by instructions in the test activity:

Test 1: The evaluator shall configure the server to send a certificate request to the client. The client shall send a certificate_list structure which has a length of zero. The evaluator shall verify that the handshake is not finished successfully and no application data flows.

Test 2: The evaluator shall configure the server to send a certificate request to the client. The client shall send no client certificate message, and instead send a client key exchange message in an attempt to continue the handshake. The evaluator shall verify that the handshake is not finished successfully and no application data flows.

Test 3: The evaluator shall configure the server to send a certificate request to the client without the supported_signature_algorithm used by the client's certificate. The evaluator shall attempt a connection using the client certificate and verify that the handshake is not finished successfully and no application data flows.

Test 4: The evaluator shall demonstrate that using a certificate without a valid certification path results in the function failing. Using the administrative guidance, the evaluator shall then load a certificate or certificates needed to validate the certificate to be used in the function, and demonstrate that the function succeeds. The evaluator then shall delete one of the certificates, and show that the function fails.

Test 5: The aim of this test is to check the response of the server when it receives a client identity certificate that is signed by an impostor CA (either Root CA or intermediate CA). To carry out this test the evaluator shall configure the client to send a client identity certificate with an issuer field that identifies a CA recognised by the TOE as a trusted CA, but where the key used for the signature on the client certificate does not in fact correspond to the CA certificate trusted by the TOE (meaning that the client certificate is invalid because its certification path does not in fact terminate in the claimed CA certificate). The evaluator shall verify that the attempted connection is denied.

Test 6: The evaluator shall configure the client to send a certificate with the Client Authentication purpose in the extendedKeyUsage field and verify that the server accepts the attempted connection. The evaluator shall repeat this test without the Client Authentication purpose and shall verify that the server denies the connection. Ideally, the two certificates should be identical except for the Client Authentication purpose.

Test 7: The evaluator shall perform the following modifications to the traffic: a) Configure the server to require mutual authentication and then modify a byte in the client's certificate. The evaluator shall verify that the server rejects the connection. b) Configure the server to require mutual authentication and then modify a byte in the signature block of the client's Certificate Verify handshake message. The evaluator shall verify that the server rejects the connection.

Test 1: The EMM AppTunnel and Push server components always send a certificate request to a client. The evaluator attempted to connect a client to each of the AppTunnel and Push TLS server implementations where the client responded to the certificate request with a certificate_list structure which had a length of zero. Using a packet capture the evaluator determined that the handshake was not finished successfully and no application data was exchanged.

Test 2: The EMM AppTunnel and Push server components always send a certificate request to a client. The evaluator attempted to connect a client to each of the AppTunnel and Push TLS server implementations where the client responded to the certificate request no client certificate message, and instead send a client key exchange message in an attempt to continue the handshake. Using a packet capture the evaluator determined that the handshake was not finished successfully and no application data was exchanged.

Test 3: The EMM AppTunnel and Push server components always require mutual authentication. The evaluator attempted to connect a client to each of the AppTunnel and Push TLS server implementations where the client to responded to the server's request for a certificate using a signature algorithm not supported by the EMM. The evaluator used a network sniffer to capture the TLS session negotiation and observed that the TLS session is rejected by the server.

Test 4: The EMM AppTunnel and Push server components always require mutual authentication. The evaluator attempted to connect a client to each of the AppTunnel and Push TLS server implementations where the client used a certificate with a broken certificate path. The evaluator used a network sniffer to capture the TLS session negotiation and observed that the TLS session is rejected by the server.

Test 5: The evaluator configured a client to send a client identity certificate to each of the AppTunnel and Push TLS server implementations with an issuer field that identifies a CA recognized by the TOE as a trusted CA, but where the key used for the signature on the client certificate does not in fact correspond to the CA certificate trusted by the TOE (i.e., the client certificate is invalid because its certification path does not in fact terminate in the claimed CA certificate). Using a packet capture the evaluator determined that the handshake was not finished successfully.

Test 6: The EMM AppTunnel and Push server components always require mutual authentication. The evaluator attempted to connect a client to each of the AppTunnel and Push TLS server implementations where the client sent a certificate with the Client Authentication purpose in the extendedKeyUsage field. Using a network sniffer to

capture the TLS session negotiation and observed that the TLS session is accepted by the server. The evaluator retried the TLS session to each of the AppTunnel and Push TLS server implementations using a client cert that is missing the Client Authentication purpose in the extendedKeyUsage field. The evaluator used a network sniffer to capture the TLS session negotiation and observed that the TLS session is rejected by the server.

Test 7: This test repeats tests FIA_X509_EXT.1.1-t5, FIA_X509_EXT.1.1-t6, and FIA_X509_EXT.1.1-t7.

2.2.19.3 PKGTLS11:FCS_TLSS_EXT.2.3

TSS Assurance Activities: If the product implements mutual authentication, the evaluator shall verify that the TSS describes how the DN and SAN in the certificate is compared to the expected identifier.

Section 6.2 of ST explains that the EMM performs hostname checking to ensure that the expected hostname the Distinguished Name (DN) in the presented certificate.

Guidance Assurance Activities: If the DN is not compared automatically to the domain name, IP address, username, or email address, the evaluator shall ensure that the AGD guidance includes configuration of the expected identifier or the directory server for the connection.

No special configuration is required to cause the EMM to perform hostname checking ensuring that the expected hostname matches the Distinguished Name (DN) in the presented certificate.

Testing Assurance Activities: Test 1: The evaluator shall send a client certificate with an identifier that does not match any of the expected identifiers and verify that the server denies the connection. The matching itself might be performed outside the TOE (e.g. when passing the certificate on to a directory server for comparison).

Using a network sniffer the evaluator captured the network traffic for the TLS client connection attempts for each of the AppTunnel and Push TLS server implementations and verified that the TLS session connected as expected using the proper certificate with matching DN. The evaluator also determined that the connection was rejected for each of the AppTunnel and Push TLS server implementations using a certificate with an DN identifier that does not match.

Component TSS Assurance Activities: None Defined

Component Guidance Assurance Activities: None Defined

Component Testing Assurance Activities: None Defined

2.2.20 TLS Server Support for Renegotiation (PKGTLS11:FCS_TLSS_EXT.4)

2.2.20.1 PKGTLS11:FCS_TLSS_EXT.4.1

TSS Assurance Activities: None Defined

Guidance Assurance Activities: None Defined

Testing Assurance Activities: None Defined

2.2.20.2 PKGTLS11:FCS_TLSS_EXT.4.2

TSS Assurance Activities: None Defined

Guidance Assurance Activities: None Defined

Testing Assurance Activities: The following tests require connection with a client that supports secure renegotiation and the 'renegotiation_info' extension.

Test 1: The evaluator shall use a network packet analyzer/sniffer to capture the traffic between the two TLS endpoints. The evaluator shall verify that the 'renegotiation_info' field is included in the ServerHello message.

Test 2: The evaluator shall modify the length portion of the field in the ClientHello message in the initial handshake to be non-zero and verify that the server sends a failure and terminates the connection. The evaluator shall verify that a properly formatted field results in a successful TLS connection.

Test 3: The evaluator shall modify the 'client_verify_data' or 'server_verify_data' value in the ClientHello message received during secure renegotiation and verify that the server terminates the connection

Note that renegotiation is supported only on the EMM Server component of the EMM which applies to the web UI and initial enrollment connections.

Test 1: The evaluator inspected the packet capture associated with the control (successful) case used with Test 2 and found that the 'renegotiation_info' field was included in the ServerHello message.

Test 2: The evaluator attempted to establish a TLS session to the EMM Server using openssl s_client with the TOE in two distinct cases: 1) the openssl library is modified to change the initial renegotiation_info extension to a non-zero extension length; and 2) an attempt is made with a properly formatted renegotiation_info extension. The former is expected to cause a failure while the latter should yield a successful connection.

Test 3: The evaluator attempted to establish a TLS session to the EMM Server using openssl s_client with the TOE where the openssl library is modified to increment the last byte of the renegotiation data by 2. The expectation is the connection will be lost upon attempting renegotiation. Note that openssl s_client is used to send "R" once connected in order to initiate the renegotiation attempt. A control test follows this test so that we have a basis of comparison.

Component TSS Assurance Activities: None Defined

Component Guidance Assurance Activities: None Defined

Component Testing Assurance Activities: None Defined

2.3 IDENTIFICATION AND AUTHENTICATION (FIA)

2.3.1 ENROLLMENT OF MOBILE DEVICE INTO MANAGEMENT (MDMPP40:FIA_ENR_EXT.1)

2.3.1.1 MDMPP40:FIA_ENR_EXT.1.1

TSS Assurance Activities: The evaluator shall examine the TSS and verify that it describes the process of enrollment for each MDM Agent/platform listed as supported in the ST. This description shall include the trusted path used for enrollment (FTP_TRP.1(2)), the method of user authentication (username/password, token, etc.), the method of authentication decision (local or remote authentication services), and the actions performed on the MDM Server upon successful authentication.

Section 6.3 of the ST states During the enrollment process, the user enters a username, mobile ID, and password as well as the EMM Server's FQDN or IP into the EMM Agent application running on their mobile device. The EMM Agent then attempts to establish an HTTPS connection with the EMM Server. The EMM Server, having authenticated to the Client through presentation of its certificate during the TLS handshake, checks that the Client/User's credentials verify correctly.

Guidance Assurance Activities: None Defined

Testing Assurance Activities: Test 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 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.

The following tests were performed using both the Android and iOS agents.

Test 1: The evaluator attempted to enroll a device without using valid credentials. The enrollment attempt failed. The evaluator examined the MDM server to ensure there is no indication of a successful enrollment.

Test 2: The evaluator attempted to enroll a device using valid credentials. The enrollment attempt succeeded. The evaluator examined the MDM server to ensure that the device appears to be properly enrolled.

2.3.1.2 MDMPP40:FIA_ENR_EXT.1.2

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

Section 6.3 of the ST states that an administrator can configure a whitelist of IMEI values such that only devices with a configured IMEI value can successfully enroll. Otherwise, an administrator can configure the EMM Server to limit users to only enrolling between one and five mobile devices.

Guidance Assurance Activities: The evaluator shall ensure that the administrative guidance describes the method(s) of restricting user enrollment and that it instructs the administrator how to configure the restrictions.

Section 11 of [Admin] the subsection entitled "Configuring the environment" describes the various configuration values available on the EMM server to control its behavior. One of these configuration values allows the administrator to specify the "Maximum Number of Active Devices per User". This value can be set to values in the range of 1 through 5, or it can be disabled (no enforced limit). Section 4 of [Admin] the subsection entitled “Managing limited enrollment” explaisn that devices can be restricted using IMEI numbers.

Testing Assurance Activities: For each type of policy selected, the evaluator shall perform the following:

Test 1: The evaluator shall attempt to configure theMDM 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).

The evaluator first configured a restriction on enrollment based on the IMEI of the device. The evaluator entered an IMEI that did not belong to any device, then attempted to enroll a device. This enrollment attempt was blocked as expected. The evaluator then replaced the invalid IMEI with the one belonging to the phone during the initial enrollment attempt, then attempted enrollment again. This time enrollment was successful based on the correct IMEI of the phone.

The evaluator then removed the IMEI restriction and configured the EMM server to limit enrollment to 1 device per user. The evaluator then attempted to exceed the configured limit by enrolling two devices. Enrollment was blocked for the 2nd device.

Component TSS Assurance Activities: None Defined

Component Guidance Assurance Activities: None Defined

Component Testing Assurance Activities: None Defined

2.3.2 Agent Enrollment of Mobile Device into Management (MDMA10:FIA_ENR_EXT.2)

2.3.2.1 MDMA10:FIA_ENR_EXT.2.1

TSS Assurance Activities: None Defined

Guidance Assurance Activities: None Defined

Testing Assurance Activities: None Defined

Component TSS Assurance Activities: The evaluator shall examine the TSS to verify that it describes which types of reference identifiers are acceptable and how the identifier is specified (e.g. preconfigured in the MDM Agent, by the user, by the MDM server, in a policy).

Section 6.3 of the ST states that during enrollment the EMM Agent records the unique URL (FQDN or IP address) of the EMM Server for future communication purposes. This value is initially configured by the mobile device user when attempting to enroll the mobile device.

Component Guidance Assurance Activities: The evaluator shall examine the operational guidance to verify that it describes how to configure reference identifier of the MDM Server's certificate and, if different than the reference identifier, the Domain Name or IP address (for connectivity) of the MDM Server.

Section 2.2 of [Install] describes the requirements for certificates used by the EMM to identify itself to other parts of the TOE and to entities in the environment. Within Section 2.2.1 of [Install], “Verifying certificate distinguished name (DN)” includes information describing the configuration constraints for the EMM regarding these component’s identification using certificate values. This includes constraints enforced on the reference identifier used by the TOE components. Section 3 of [Install] also includes instructions for the installation of the EMM.

Component Testing Assurance Activities: The evaluator shall follow the operational guidance to establish the reference identifier of the MDM server on the MDM Agent and in conjunction with other evaluation activities verify that the MDM Agent can connect to the MDM Server and validate the MDM Server's certificate.

This test case repeats the test described in FIA_ENR_EXT.1.1-t2.

2.3.3 Timing of Authentication (MDMPP40:FIA_UAU.1)

2.3.3.1 MDMPP40:FIA_UAU.1.1

TSS Assurance Activities: 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 requirement selects implement for this requirement.

Guidance Assurance Activities: None Defined

Testing Assurance Activities: None Defined

2.3.3.2 MDMPP40:FIA_UAU.1.2

TSS Assurance Activities: 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 requirement selects implement for this requirement

Guidance Assurance Activities: None Defined

Testing Assurance Activities: The evaluator shall perform the following tests:

Test 1: The evaluator shall attempt to perform the prohibited actions before authentication. The evaluator shall verify the actions cannot be performed.

Test 2: The evaluator shall attempt to perform the prohibited actions after authentication. The evaluator shall verify the actions can be performed.

Test 1: The evaluator attempted the actions that are available through the EMM server interface prior to login and found only those claimed by the SFR were available.

Test 2: The evaluator confirmed that only authorized administrators can login to the TOE. Thus there are no prohibited actions after authentication.

Component TSS Assurance Activities: None Defined

Component Guidance Assurance Activities: None Defined

Component Testing Assurance Activities: None Defined

2.3.4 X.509 Certificate Validation - per TD0641 (MDMPP40:FIA_X509_EXT.1(1))

2.3.4.1 MDMPP40:FIA_X509_EXT.1.1(1)

TSS Assurance Activities: 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.

Section 6.3 of ST indicates that the TOE selected both "invoke platform-provided functionality" and "implement functionality" for its processing of x509 certificate validation. The EMM performs certificate validity checking while the EMM Agent uses the underlying phone to perform certificate validity checking of the peer's certificate and certificate chain.

The TOE validates authentication certificates (including the full path) and checks their revocation station using CRLs and OCSP. The TOE processes certificates presented during the TLS handshake by first checking the received certificate's validity period and appropriate key usage property. The TOE checks that it can construct a certificate path from the peer's certificate through any intermediary CAs to a trusted root CA. If the TOE can successfully build the certificate path, then the TOE will next check the validity of the CA certificates (e.g., checking its validity dates and that the CA flag is present in the basic constraints section for all CA certs) in the chain. Assuming the TOE determines that all CA certificates in the chain are valid, the TOE will finally check the revocation status of the server's certificate. The TOE will accept any certificate for which it cannot determine the revocation status and will accept the connection attempt.

The EMM Agent components receive unique certificates during their enrollment process and they store the received certificates in the Android keystore (which stores the keys with permissions to only allow the applications themselves to access the keys). When the EMM Agent components subsequently contact the EMM, it will utilize the keys in the keystore. Each of the EMM Agent components stores a single key in the Android keystore and does not attempt to store multiple keys. When a Client's keys expire, the user must un-enroll (which destroys the Client

component certificates in Android's keystore) and re-enroll their device to obtain new certificates (which the components load again into the keystore).

Guidance Assurance Activities: If 'internal lookup of TOE-managed certificate status' is selected, then the evaluator shall ensure the AGD documentation describes how issued certificates are reported as invalid.

Internal lookup was not selected.

Testing Assurance Activities: 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 four certificates: the node certificate to be tested, two Intermediate CAs, and the self-signed Root CA.

Test 1: The evaluator shall demonstrate that validating a certificate without a valid certification path results in the function failing, for each of the following reasons, in turn:

- by establishing a certificate path in which one of the issuing certificates is not a CA certificate,

- by omitting the basicConstraints field in one of the issuing certificates,

- by setting the basicConstraints field in an issuing certificate to have CA=False,

- by omitting the CA signing bit of the key usage field in an issuing certificate, and

- by setting the path length field of a valid CA field to a value strictly less than the certificate path.

The evaluator shall then establish a valid certificate path consisting of valid CA certificates, and demonstrate that the function succeeds. The evaluator shall then remove trust in one of the CA certificates, and show that the function fails.

Test 2: The evaluator shall demonstrate that validating an expired certificate results in the function failing.

Test 3: The evaluator shall test that the TOE can properly handle revoked certificates - conditional on whether CRL, OCSP, or OCSP stapling, or certificate status lookup 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. For FIA_X509_EXT.1.1(2) if included, if 'no revocation method' is selected, this test is omitted. The evaluator shall ensure that a valid certificate is used, and that the validation function succeeds. The evaluator then attempts the test with a certificate that has been revoked (for each method chosen in the selection) to ensure when the certificate is no longer valid that the validation function fails, if 'internal lookup of TOE-managed certificate status' is selected, then the evaluator shall follow AGD guidance to report the certificate as invalid.

Test 4: [conditional] 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. If certificate status lookup is selected, this test is omitted. For FIA_X509_EXT.1.1(2) if included, if 'no revocation method' is selected, this test is omitted.

Test 5: The evaluator shall modify any byte in the first eight bytes of the certificate and demonstrate that the certificate fails to validate. (The certificate will fail to parse correctly).

Test 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 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 8a: (Conditional on support for EC certificates as indicated in FCS_COP.1(3)). 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(3)). 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.

Note that since the EMM Agent utilizes its host device for TLS and X509 functions, it is not subject to these test cases. However, this testing applies to both client and server implementations of the EMM where mutual authentication is performed – as such the tests have been performed for the AppTunnel client and server implementations and Push client and server implementations.

Test 1: The evaluator attempted to successfully connect to each of the AppTunnel TLS server and Push TLS server implementations from a test client using a good client certificate as a baseline and alternately connect from each of the AppTunnel TLS client and Push TLS client implementations to a test server using a good server certificate as a baseline. The evaluator used a network sniffer to capture the TLS session negotiations and observed the connection was successful. The evaluator then configured a server certificate with an invalid certification path by deleting the root CA so that the certificate chain was invalid because of a missing (or deleted) certificate. The connection was refused in each case. The evaluator then configured a test server to send an authentication certificate issued by a Sub CA with no BasicConstraints and with BasicConstraints but the CA Flag set to false and then with an intermediate CA with no keyCertSign purpose and then with an intermediate CA with a path length field set too low. The connection was refused in each case.

Test 2: The evaluator attempted to make a connection to each of the AppTunnel TLS server and Push TLS server implementations from a test client using an expired client certificate and alternately connect from each of the AppTunnel TLS client and Push TLS client implementations to a test server using an expired server certificate as a baseline. The evaluator used a network sniffer to capture the TLS session negotiations and observed the connection was not successful. The evaluator then configured a server certificate that had an expired subCA. The connection was refused in each case.

Test 3: The evaluator used a test client to attempt to connect to each of the AppTunnel TLS server and Push TLS server implementations and alternately connect from each of the AppTunnel TLS client and Push TLS client implementations to a test server. The test client/server presented a certificate during the TLS negotiation where the certificate was valid. A packet capture was obtained of this TLS negotiation which shows that the connection was successful. The evaluator revoked the test client/server certificate and attempted the same connection from the test client. The connection attempt performed after revoking the certificate was not successful.

Test 4: For the CRL, the evaluator configured a CRL provider to issue a CRL that was signed by a certificate that did not have the CRLsign Key usage bit set. The evaluator then attempted to connect to each of the AppTunnel TLS server and Push TLS server implementations from a test client where the certificate just described was used by the test client and alternately connect from each of the AppTunnel TLS client and Push TLS client implementations to a test server where the certificate just described was used by the test server. The evaluator also attempted to connect to a test client/server for each implementation identified above where the test client/server was using the improper (an untrusted certificate) CRLsign certificate. All connection attempts failed with the TOE detecting the improper certificate being used to sign the CRL. This was repeated for OCSP and OCSP signing purpose with comparable results.

Test 5: The evaluator configured the test client to send an authentication certificate 1) that is valid, 2) that has one byte in the ASN1 field changed, 3) that has one byte in the certificate signature changed, and 4) that has one byte in the certificate public key changed, and attempted to make a connection from the test client to each of the AppTunnel TLS server and Push TLS server implementations and alternately connect from each of the AppTunnel TLS client and Push TLS client implementations to a test server using that certificate. The evaluator used a network sniffer to capture the TLS session negotiations and observed each connection was not successful.

Test 6 – This test was performed with test 5.

Test 7 – This test was performed with test 5.

Test 8.1 – The evaluator configured valid server and client certificates. A successful connection was made in each case. The evaluator then configured the server to send an authentication certificate with an explicitly defined elliptic curve. The connection was refused in each case.

Test 8.2 – This was tested with 8.1.

2.3.4.2 MDMPP40:FIA_X509_EXT.1.2(1)

TSS Assurance Activities: 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).

Section 6.3 of ST indicates that the TOE selected both "invoke platform-provided functionality" and "implement functionality" for its processing of x509 certificate validation. The EMM performs certificate validity checking while the EMM Agent uses the underlying phone to perform certificate validity checking of the peer's certificate and certificate chain.

The EMM Agent components receive unique certificates during their enrollment process and they store the received certificates in the Android keystore (which stores the keys with permissions to only allow the applications themselves to access the keys). When the EMM Agent components subsequently contact the EMM, it will utilize the keys in the keystore. Each of the EMM Agent components stores a single key in the Android keystore and does not attempt to store multiple keys. When a Client's keys expire, the user must un-enroll (which destroys the Client component certificates in Android's keystore) and re-enroll their device to obtain new certificates (which the components load again into the keystore).

Guidance Assurance Activities: None Defined

Testing Assurance Activities: 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 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 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 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.

Note that since the EMM Agent utilizes its host device for TLS and X509 functions, it is not subject to these test cases. However, this testing applies to both client and server implementations of the EMM where mutual authentication is performed – as such the tests have been performed for the AppTunnel client and server implementations and Push client and server implementations.

Test 1: The evaluator configured an issuing CA using a certificate without the basicConstraints extension. The evaluator then attempted to connect from a test client to each of the AppTunnel TLS server and Push TLS server implementations and alternately connect from each of the AppTunnel TLS client and Push TLS client implementations to a test server where the certificate used by the test server signed by the CA certificate just described. All connection attempts failed with the EMM detecting the improper certificate path.

Test 2: The evaluator configured an issuing CA using a certificate with the basicConstraints extension with the cA flag that is not set. The evaluator then attempted to connect from a test client to each of the AppTunnel TLS server and Push TLS server implementations and alternately connect from each of the AppTunnel TLS client and Push TLS client implementations to a test server where the certificate used by the test server signed by the CA certificate was as just described. All connection attempts failed with the EMM detecting the improper certificate path.

Test 3: The evaluator configured an issuing CA using a certificate with the basicConstraints extension with the cA flag that is set as TRUE. The evaluator then attempted to connect from a test client to each of the AppTunnel TLS server and Push TLS server implementations and alternately connect from each of the AppTunnel TLS client and Push TLS client implementations to a test server where the certificate used by the test server signed by the CA certificate was as just described. All connection attempts succeeded with the EMM accepting the certificate path.

Component TSS Assurance Activities: None Defined

Component Guidance Assurance Activities: None Defined

Component Testing Assurance Activities: None Defined

2.3.5 X.509 Certificate Validation - per TD0641 (MDMPP40:FIA_X509_EXT.1(2))

2.3.5.1 MDMPP40:FIA_X509_EXT.1.1(2)

TSS Assurance Activities: 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 funcationality 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.

Section 6.3 of ST indicates that the TOE selected both "invoke platform-provided functionality" and "implement functionality" for its processing of x509 certificate validation. The EMM performs certificate validity checking while the EMM Agent uses the underlying phone to perform certificate validity checking of the peer's certificate and certificate chain. The EMM and EMM Agent use X.509 certificates only during TLS/HTTPS trusted channel establishment (for server and client/mutual authentication). The EMM and EMM Agent use X.509v3 certificates for TLS authentication and will not establish a TLS session if the certificate presented by the peer is determined to be invalid.

The TOE validates authentication certificates (including the full path) and checks their revocation station using CRLs and OCSP. The TOE processes certificates presented during the TLS handshake by first checking the received certificate's validity period and appropriate key usage property. The TOE checks that it can construct a certificate path from the peer's certificate through any intermediary CAs to a trusted root CA. If the TOE can successfully build the certificate path, then the TOE will next check the validity of the CA certificates (e.g., checking its validity dates and that the CA flag is present in the basic constraints section for all CA certs) in the chain. Assuming the TOE determines that all CA certificates in the chain are valid, the TOE will finally check the revocation status of the server's certificate. The TOE will accept any certificate for which it cannot determine the revocation status and will accept the connection attempt.

The EMM Agent components receive unique certificates during their enrollment process and they store the received certificates in the Android keystore (which stores the keys with permissions to only allow the applications themselves to access the keys). When the EMM Agent components subsequently contact the EMM, it will utilize the keys in the keystore. Each of the EMM Agent components stores a single key in the Android keystore and does not attempt to store multiple keys. When a Client's keys expire, the user must un-enroll (which destroys the Client component certificates in Android's keystore) and re-enroll their device to obtain new certificates (which the components load again into the keystore).

Guidance Assurance Activities: If 'internal lookup of TOE-managed certificate status' is selected, then the evaluator shall ensure the AGD documentation describes how issued certificates are reported as invalid.

Internal lookup as not selected.

Testing Assurance Activities: 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 four certificates: the node certificate to be tested, two Intermediate CAs, and the self-signed Root CA.

Test 1: The evaluator shall demonstrate that validating a certificate without a valid certification path results in the function failing, for each of the following reasons, in turn:

- by establishing a certificate path in which one of the issuing certificates is not a CA certificate,

- by omitting the basicConstraints field in one of the issuing certificates,

- by setting the basicConstraints field in an issuing certificate to have CA=False,

- by omitting the CA signing bit of the key usage field in an issuing certificate, and

- by setting the path length field of a valid CA field to a value strictly less than the certificate path.

The evaluator shall then establish a valid certificate path consisting of valid CA certificates, and demonstrate that the function succeeds. The evaluator shall then remove trust in one of the CA certificates, and show that the function fails.

Test 2: The evaluator shall demonstrate that validating an expired certificate results in the function failing.

Test 3: The evaluator shall test that the TOE can properly handle revoked certificates - conditional on whether CRL, OCSP, or OCSP stapling, or certificate status lookup 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. For FIA_X509_EXT.1.1(2) if included, if 'no revocation method' is selected, this test is omitted. The evaluator shall ensure that a valid certificate is used, and that the validation function succeeds. The evaluator then attempts the test with a certificate that has been revoked (for each method chosen in the selection) to ensure when the certificate is no longer valid that the validation function fails, if 'internal lookup of TOE-managed certificate status' is selected, then the evaluator shall follow AGD guidance to report the certificate as invalid.

Test 4: [conditional] 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. If certificate status lookup is selected, this test is omitted. For FIA_X509_EXT.1.1(2) if included, if 'no revocation method' is selected, this test is omitted.

Test 5: The evaluator shall modify any byte in the first eight bytes of the certificate and demonstrate that the certificate fails to validate. (The certificate will fail to parse correctly).

Test 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 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 8a: (Conditional on support for EC certificates as indicated in FCS_COP.1(3)). 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(3)). 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.

Note that since the EMM Agent utilizes its host device for TLS and X509 functions, it is not subject to these test cases. However, this testing applies to both client and server implementations of the EMM where mutual authentication is performed – as such the tests have been performed for the AppTunnel client and server implementations and Push client and server implementations.

Test 1: The evaluator attempted to successfully connect to each of the AppTunnel TLS server and Push TLS server implementations from a test client using a good client certificate as a baseline and alternately connect from each of the AppTunnel TLS client and Push TLS client implementations to a test server using a good server certificate as a baseline. The evaluator used a network sniffer to capture the TLS session negotiations and observed the connection was successful. The evaluator then configured a server certificate with an invalid certification path by deleting the root CA so that the certificate chain was invalid because of a missing (or deleted) certificate. The connection was refused in each case. The evaluator then configured a test server to send an authentication certificate issued by a Sub CA with no BasicConstraints and with BasicConstraints but the CA Flag set to false and then with an intermediate CA with no keyCertSign purpose and then with an intermediate CA with a path length field set too low. The connection was refused in each case.

Test 2: The evaluator attempted to make a connection to each of the AppTunnel TLS server and Push TLS server implementations from a test client using an expired client certificate and alternately connect from each of the AppTunnel TLS client and Push TLS client implementations to a test server using an expired server certificate as a baseline. The evaluator used a network sniffer to capture the TLS session negotiations and observed the connection was not successful. The evaluator then configured a server certificate that had an expired subCA. The connection was refused in each case.

Test 3: The evaluator used a test client to attempt to connect to each of the AppTunnel TLS server and Push TLS server implementations and alternately connect from each of the AppTunnel TLS client and Push TLS client

implementations to a test server. The test client/server presented a certificate during the TLS negotiation where the certificate was valid. A packet capture was obtained of this TLS negotiation which shows that the connection was successful. The evaluator revoked the test client/server certificate and attempted the same connection from the test client. The connection attempt performed after revoking the certificate was not successful.

Test 4: For the CRL, the evaluator configured a CRL provider to issue a CRL that was signed by a certificate that did not have the CRLsign Key usage bit set. The evaluator then attempted to connect to each of the AppTunnel TLS server and Push TLS server implementations from a test client where the certificate just described was used by the test client and alternately connect from each of the AppTunnel TLS client and Push TLS client implementations to a test server where the certificate just described was used by the test server. The evaluator also attempted to connect to a test client/server for each implementation identified above where the test client/server was using the improper (an untrusted certificate) CRLsign certificate. All connection attempts failed with the TOE detecting the improper certificate being used to sign the CRL. This was repeated for OCSP and OCSP signing purpose with comparable results.

Test 5: The evaluator configured the test client to send an authentication certificate 1) that is valid, 2) that has one byte in the ASN1 field changed, 3) that has one byte in the certificate signature changed, and 4) that has one byte in the certificate public key changed, and attempted to make a connection from the test client to each of the AppTunnel TLS server and Push TLS server implementations and alternately connect from each of the AppTunnel TLS client and Push TLS client implementations to a test server using that certificate. The evaluator used a network sniffer to capture the TLS session negotiations and observed each connection was not successful.

Test 6 – This test was performed with test 5.

Test 7 – This test was performed with test 5.

Test 8.1 – The evaluator configured valid server and client certificates. A successful connection was made in each case. The evaluator then configured the server to send an authentication certificate with an explicitly defined elliptic curve. The connection was refused in each case.

Test 8.2 – This was tested with 8.1.

2.3.5.2 MDMPP40:FIA_X509_EXT.1.2(2)

TSS Assurance Activities: 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).

See MDMPP40:FIA_X509_EXT.1.1(2)

Guidance Assurance Activities: None Defined

Testing Assurance Activities: 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 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 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 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.

Note that since the EMM Agent utilizes its host device for TLS and X509 functions, it is not subject to these test cases. However, this testing applies to both client and server implementations of the EMM where mutual authentication is performed – as such the tests have been performed for the AppTunnel client and server implementations and Push client and server implementations.

Test 1: The evaluator configured an issuing CA using a certificate without the basicConstraints extension. The evaluator then attempted to connect from a test client to each of the AppTunnel TLS server and Push TLS server implementations and alternately connect from each of the AppTunnel TLS client and Push TLS client implementations to a test server where the certificate used by the test server signed by the CA certificate just described. All connection attempts failed with the EMM detecting the improper certificate path.

Test 2: The evaluator configured an issuing CA using a certificate with the basicConstraints extension with the cA flag that is not set. The evaluator then attempted to connect from a test client to each of the AppTunnel TLS server and Push TLS server implementations and alternately connect from each of the AppTunnel TLS client and Push TLS client implementations to a test server where the certificate used by the test server signed by the CA certificate was as just described. All connection attempts failed with the EMM detecting the improper certificate path.

Test 3: The evaluator configured an issuing CA using a certificate with the basicConstraints extension with the cA flag that is set as TRUE. The evaluator then attempted to connect from a test client to each of the AppTunnel TLS server and Push TLS server implementations and alternately connect from each of the AppTunnel TLS client and Push TLS client implementations to a test server where the certificate used by the test server signed by the CA certificate was as just described. All connection attempts succeeded with the EMM accepting the certificate path.

Component TSS Assurance Activities: None Defined

Component Guidance Assurance Activities: None Defined

Component Testing Assurance Activities: None Defined

2.3.6 X.509 Certificate Authentication - per TD0641 (MDMPP40:FIA_X509_EXT.2)

2.3.6.1 MDMPP40:FIA_X509_EXT.2.1

TSS Assurance Activities: 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).

Section 6.3 of ST indicates that the TOE selected both "invoke platform-provided functionality" and "implement functionality" for its processing of x509 certificate validation. The EMM performs certificate validity checking while the EMM Agent uses the underlying phone to perform certificate validity checking of the peer's certificate and certificate chain.

The TOE validates authentication certificates (including the full path) and checks their revocation station using CRLs and OCSP. The TOE processes certificates presented during the TLS handshake by first checking the received certificate's validity period and appropriate key usage property. The TOE checks that it can construct a certificate path from the peer's certificate through any intermediary CAs to a trusted root CA. If the TOE can successfully build the certificate path, then the TOE will next check the validity of the CA certificates (e.g., checking its validity dates and that the CA flag is present in the basic constraints section for all CA certs) in the chain. Assuming the TOE determines that all CA certificates in the chain are valid, the TOE will finally check the revocation status of the server's certificate. The TOE will accept any certificate for which it cannot determine the revocation status and will accept the connection attempt.

The EMM Agent components receive unique certificates during their enrollment process and they store the received certificates in the Android keystore (which stores the keys with permissions to only allow the applications themselves to access the keys). When the EMM Agent components subsequently contact the EMM, it will utilize the keys in the keystore. Each of the EMM Agent components stores a single key in the Android keystore and does not attempt to store multiple keys. When a Client's keys expire, the user must un-enroll (which destroys the Client component certificates in Android's keystore) and re-enroll their device to obtain new certificates (which the components load again into the keystore).

Guidance Assurance Activities: None Defined

Testing Assurance Activities: None Defined

2.3.6.2 MDMPP40:FIA_X509_EXT.2.2

TSS Assurance Activities: 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.

Section 6.3 of ST describes that the EMM Agent components receive unique certificates during their enrollment process and they store the received certificates in the Android keystore (which stores the keys with permissions to only allow the applications themselves to access the keys). The EMM server components receive certificates during the installation process, and each component has a single certificate for each TLS port enabled, thus each component offering a TLS connection (whether acting as a TLS server or client) always uses its single, configured certificate.

Section 6.3 of ST describes that the Server performs validity checking while the Agent uses the underlying phone to perform certificate validity checking of the server's certificate and certificate chain.

Section 6.3 of ST states that both the Server and Agent will accept as valid any certificate for which the Server or Agent cannot reach the revocation server to check its status

Guidance Assurance Activities: 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.

Both the Server and Agent will accept as valid any certificate for which the Server or Agent cannot reach the revocation server to check its status. This is not a configurable setting.

Testing Assurance Activities: The evaluator shall perform the following test for each trusted channel:

Test 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.

The evaluator performed successful connections using valid certificates as part of the FIA_X509_EXT.1.1 and FIA_X509_EXT.1(1).1 test cases.

The evaluator attempted a connection where the certificates sent by the peer contained revocation information for a revocation server that could not be contacted. The TOE accepted this connection, so the evaluator concluded that the TOE accepts connections when the revocation server cannot be contacted.

Component TSS Assurance Activities: None Defined

Component Guidance Assurance Activities: None Defined

Component Testing Assurance Activities: None Defined

2.3.7 X.509 Unique Certificate (MDMPP40:FIA_X509_EXT.5)

2.3.7.1 MDMPP40:FIA_X509_EXT.5.1

TSS Assurance Activities: None Defined

Guidance Assurance Activities: None Defined

Testing Assurance Activities: None Defined

Component TSS Assurance Activities: 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 then the evaluator shall examine the TSS to verify that it describes the methods to ensure that each client utilizes a unique certificate.

Section 6.3 of ST describes that the EMM Agent components receive unique certificates during their enrollment process and they store the received certificates in the Android keystore (which stores the keys with permissions to only allow the applications themselves to access the keys). The EMM server components receive certificates during the installation process, and each component has a single certificate for each TLS port enabled, thus each component offering a TLS connection (whether acting as a TLS server or client) always uses its single, configured certificate.

Component Guidance Assurance Activities: None Defined

Component Testing Assurance Activities: For each MDM Agent/platform listed as supported in the ST:

The evaluator shall utilize 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.

One of the following tests must be performed depending on whether the MDM agent allows for the loading of certificates.

Test 1: [conditional] If the MDM agent allows for the loading of certificates: The evaluator shall initiate communications between the MDM Server and a client device over a trusted channel established using the device's unique certificate, verifying that a successful communication channel was established. The evaluator shall then attempt to initiate communications between the MDM Server and a second client device over a trusted channel established using the unique certificate from the first device, verifying that the MDM Server rejects this attempt at communication.

Test 2: [conditional] If the MDM agent does not allow for the loading of certificates: The evaluator shall concurrently enroll 10 devices and ensure that the client certificate for each is unique, per the methods described in the TSS.

Test 1: [conditional] The TOE loads certificates during enrollment. Thus this test is not applicable.

Test 2: [conditional] The evaluator enrolled ten (10) devices with the EMM and inspected the certificates on each device using device specific debugging tools. The evaluator found that each device had a certificate with a Common Name that was unique from the others.

2.4 Security management (FMT)

2.4.1 Management of Functions Behavior (MDMPP40:FMT_MOF.1(1))

2.4.1.1 MDMPP40:FMT_MOF.1.1(1)

TSS Assurance Activities: None Defined

Guidance Assurance Activities: None Defined

Testing Assurance Activities: None Defined

Component TSS Assurance Activities: 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.

Section 6.4 of ST indicates that the EMM restricts all security management functions to an authorized administrator. It provides a table identifying the management functions available to an administrator to configure profiles for agents on Android and Apple mobile devices. These management functions cover the commands, configurations, and policies that can be sent to a mobile device.

Component Guidance Assurance Activities: None Defined

Component Testing Assurance Activities: Test 1: The evaluator shall attempt to access the functions and policies in FMT_SMF.1(1) as an unauthorized user and verify that the attempt fails.

Test 2: [conditional] The evaluator shall attempt to access the functions and policies in FMT_SMF.1(3) as an unauthorized user and verify that the attempt fails.

Refer to test case FMT_SMF.1(3) where the evaluator attempted to exercise the functions as an unauthorized administrator.

2.4.2 Management of Functions Behavior (Enrollment) (MDMPP40:FMT_MOF.1(2))

2.4.2.1 MDMPP40:FMT_MOF.1.1(2)

TSS Assurance Activities: None Defined

Guidance Assurance Activities: None Defined

Testing Assurance Activities: None Defined

Component TSS Assurance Activities: The evaluator shall examine the TSS and verify that it describes how unauthorized users are prevented from enrolling in the MDM services.

Section 6.4 of ST explains that an EMM administrator can enable mobile device users to enroll their mobile device. This is done by having the administrator provide the mobile device user with a username and a password that will allow them to enroll their devices.

Component Guidance Assurance Activities: None Defined

Component Testing Assurance Activities: The test of this function is performed in conjunction with FIA_ENR_EXT.1.

See the test case for MDMPP40:FIA_ENR_EXT.1.

2.4.3 MANAGEMENT OF FUNCTIONS IN (MAS SERVER DOWNLOADS) (MDMPP40:FMT_MOF.1(3))

2.4.3.1 MDMPP40:FMT_MOF.1.1(3)

TSS Assurance Activities: None Defined

Guidance Assurance Activities: None Defined

Testing Assurance Activities: None Defined

Component TSS Assurance Activities: The evaluator shall examine the TSS to determine that all methods of initiating an application download or update push are specified.

The MDM and MAS server for this TOE are the same. Section 6.4 of ST indicates that the administrator can define device groups which allows for the grouping of mobile devices having the same device management profile and app management profile.

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

Section 3 of [Admin] the subsection entitled "Managing groups" describes the EMM's Group feature. This can be used to define a group of users or mobile devices, and to associate a profile (policies) and applications with the group.

Component Testing Assurance Activities: The evaluator shall ensure that the MAS Server verifies that the mobile device is enrolled in the MDM Server and is in a compliant state. The evaluator shall verify that an application cannot be downloaded from the MAS Server prior to enrolling the device with the MDM. The evaluator shall partially enroll the mobile device, so the device is connected to the MDM Server, but is not compliant and verify that applications cannot be downloaded.

The evaluator installed multiple Android apps into the MAS server and created two App Management Profiles (one for each app). The evaluator assigned one profile to a mobile device and inspected the apps available through the MAS server on that device. Only the one application assigned to the profile was visible through the MAS server. The evaluator then assigned a second App management profile (which specified a different application) to the mobile device. The evaluator checked the applications visible from the MAS server and observed that the new application was visible and the old was no longer available for installation.

The evaluator used an enrolled device to display the applications visible to the mobile device through the MAS server. From the EMM, the evaluator un-enrolled the mobile device at the EMM Server and tried to install an application on the mobile device. The evaluator found that visibility into the MAS server's install applications was lost following the device being un-enrolled.

2.4.4 Trusted Policy Update (MDMPP40:FMT_POL_EXT.1)

2.4.4.1 MDMPP40:FMT_POL_EXT.1.1

TSS Assurance Activities: None Defined

Guidance Assurance Activities: None Defined

Testing Assurance Activities: None Defined

Component TSS Assurance Activities: Policies must be digitally signed by the enterprise using the algorithms in FCS_COP.1(3). The evaluator shall ensure that the TSS describes how policies are digitally signed by the TSF.

Section 6.4 of ST indicates that the EMM Server signs each policy with a certificate configured for that purpose that the EMM Agent is configured to trust upon enrollment. When the EMM Agent receives any policy, it checks its signature and if the check fails the policy is discarded and an alert is sent to the EMM.

Component Guidance Assurance Activities: If applicable, the evaluator shall verify that the AGD guidance instructs administrators on configuring the Enterprise certificate to be used for signing policies or signing the policies before applying them.

The TOE does not require special configuration actions for the MDM Agent to trust the certificates used to sign policies.

Component Testing Assurance Activities: The evaluator shall perform a policy update in accordance with FMT_SMF.1(1). The evaluator shall examine the policy either at the MDM Server, in transmission, or at the MDM agent, and verify theTSF signs the update and provides it to the MDM Agent.

The evaluator loaded a TOE modification that displayed the signature and policy of a policy update being pushed to a device as it about to be sent. Inspection of this evidence showed that the policy was signed.

2.4.5 Agent Trusted Policy Update (MDMA10:FMT_POL_EXT.2)

2.4.5.1 MDMA10:FMT_POL_EXT.2.1

TSS Assurance Activities: None Defined

Guidance Assurance Activities: None Defined

Testing Assurance Activities: None Defined

2.4.5.2 MDMA10:FMT_POL_EXT.2.2

TSS Assurance Activities: None Defined

Guidance Assurance Activities: None Defined

Testing Assurance Activities: None Defined

Component TSS Assurance Activities: The evaluator ensures that the TSS describes how the candidate policies are obtained by the MDM Agent, the processing associated with verifying the digital signature of the policy updates, and the actions that take place for successful (signature was verified) and unsuccessful (signature could not be verified) cases. The software components that are performing the processing must also be identified in the TSS and verified by the evaluators.

Section 6.4 of ST explains that the EMM Server signs each policy with a certificate configured for that purpose that the EMM Agent is configured to trust upon enrollment. When the EMM Agent receives any policy, it checks its signature and if the check fails the policy is discarded and an alert is sent to the EMM.

Component Guidance Assurance Activities: None Defined

Component Testing Assurance Activities: This evaluation activity is performed in conjunction with the evaluation activity for FIA_X509_EXT.1 and FIA_X509_EXT.2 as defined in the Base-PPs.

Test 1: The evaluator shall perform a policy update from an available configuration interface (such as through a test MDM Server). The evaluator shall verify the update is signed and is provided to the MDM Agent. The evaluator shall verify the MDM Agent accepts the digitally signed policy.

Test 2: The evaluator shall perform a policy update from an available configuration interface (such as through a test MDM Server). The evaluator shall provide an unsigned and an incorrectly signed policy to the MDM Agent. The evaluator shall verify the MDM Agent does not accept the digitally signed policy.

Test 1: The evaluator used the EMM Server to push a valid, signed policy to a device and observed the MDM Agent accepted and applied the policy.

Test 2: The evaluator loaded custom code provided by the vendor into the EMM that placed invalid signatures on policy updates. The evaluator then attempted to push a an otherwise valid policy update to the device and observed that the MDM Agent rejected the policy update. The evaluator repeated these steps after loading custom code provided by the vendor that did not sign the updates at all. When no signature was applied to the policy update, the MDM Agent rejected the policy update.

2.4.6 Specification of Management Functions (Server configuration of Agent) (MDMPP40:FMT_SMF.1(1))

2.4.6.1 MDMPP40:FMT_SMF.1.1(1)

TSS Assurance Activities: None Defined

Guidance Assurance Activities: None Defined

Testing Assurance Activities: None Defined

Component TSS Assurance Activities: 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/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/platform are clearly indicated.

The evaluator shall determine if the Mobile Device 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 Mobile Device and which management functions were not evaluated.

(TD0479 applied)

Section 6.4 of ST includes a table that describes the management functions available on the EMM Server. That table identifies the functions which are implemented by the "Android agent" and those which are implemented by the "Apple agent". The claims regarding management capabilities made by the table in Section 6.4 were exercised during testing and found to be consistent with the Apple and Android mobile device's capabilities.

Component Guidance Assurance Activities: None Defined

Component Testing Assurance Activities: For each MDM Agent/platform listed as supported in the ST:

Test 1: The evaluator shall verify the ability to command eachMDM Agent functional capability and configure each MDM Agent policy listed above.

For commands, the evaluator issued each claimed command iteratively from the EMM Server to an enrolled device and observed that the corresponding action occurred on the mobile device. These commands were sent to both the EMM Android agent and the iOS agent.

The evaluator tested the application of policies for both the EMM Android agent and the iOS agent. For policies, the evaluator crafted a policy that included settings for each claimed policy that were known to be different from default settings. The evaluator pushed the policy to an enrolled device and then performed functions on the mobile device to show that the settings were applied and effective. The evaluator modified the policy to have

different settings in each case. The evaluator pushed the revised policy to an enrolled device and again performed functions on the mobile device to show that the settings were applied and effective as expected.

2.4.7 Specification of Management Functions (Server Configuration of Server) (MDMPP40:FMT_SMF.1(2))

2.4.7.1 MDMPP40:FMT_SMF.1.1(2)

TSS Assurance Activities: None Defined

Guidance Assurance Activities: None Defined

Testing Assurance Activities: None Defined

Component TSS Assurance Activities: The evaluator shall examine the TSS to ensure that it describes each management function listed. For function c.4, 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 mobile devices.

Section 6.4 of ST indicates that the EMM Server component of the TOE supports the security management functions to configure and manage itself, including configuring a login(unlock) banner. The claimed functions in this section encompass exactly the same set of management functions identified by FMT_SMF.1(2).1. Function c.4 was not claimed.

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

Section 6 of [Admin] the section entitled "Configuring policies by device platform" explains how to configure a policy to be applied to a mobile device. Section 3 of [Admin] the subsection entitled "Managing groups" describes how to assign profiles to groups. Finally, Section 6 of [Admin] the section entitled "Creating Profiles" provides a detailed description of all of the policy values that can be applied on mobile devices using the profiles.

Reference identifiers used within certificates are not configurable by users or administrators. Users and administrators use a Mobile ID which is simply a user given name for the device. This name is automatically mapped by the system to the reference identifier used within the certificate that identifies the mobile device.

Component Testing Assurance Activities: The tests of functions b, c.1, c.2, and c.5 are performed in conjunction with the use of the function. Test 3 also covers function c.4. The evaluator shall perform the following test:

Test 1: The evaluator shall configure the TSF authentication certificate(s) and verify that the correct certificate is used in established trusted connections (FPT_ITT.1(1), FPT_ITT.1(2), FTP_ITC.1(1), and FTP_TRP.1(2)).

Test 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 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.

Test 1: During testing associated with FPT_ITT.1, FTP_ITC.1 and FTP_TRP.1, the evaluator configured certificates to be used by the TOE, and observed that those certificates were used during TLS negotiation.

Test 2: The evaluator configured an "Interval between MDM Agent Pollings" of 1 hour for Android devices. The evaluator then waited approximately 3.5 hours and examined the audit logs. The audits showed that the server polled the agent approximately every hour. The evaluator then set the polling interval at 4 hours and observed that polling occurred every 4 hours.

Test 3: The evaluator configured X.509v3 certs for use on the MDM server, the number of devices allowed for enrollment per user, and the TOE unlock banner. The evaluator observed that all configured values were present in the MD after the policy had been updated.

2.4.8 Specification of Management Functions (MAS Server) (MDMPP40:FMT_SMF.1(3))

2.4.8.1 MDMPP40:FMT_SMF.1.1(3)

TSS Assurance Activities: None Defined

Guidance Assurance Activities: None Defined

Testing Assurance Activities: None Defined

Component TSS Assurance Activities: 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.

Section 6.4 of the ST states the EMM Server provides the MAS server functionality. Furthermore, in support of application hosting, the MDM server supports the configuration of application groups assigned to individual apps and devices. It also supports the ability to download applications for deployment.

Component Guidance Assurance Activities: 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 AGD guidance includes detailed instructions of what options are available and how to configure each management functional capability listed.

Section 5 of [Admin] the section entitled "Applications" describes how to manage applications using the MAS feature of the EMM. Internal applications are the applications provided to users of mobile devices through the MAS feature of the EMM. Public applications are the applications that can be purchased from the Google Play Store or Apple App Store. Administrators can activate or deactivate Internal and Public applications in the admin console to control download and execution of applications on mobile devices.

Section 3 of [Admin] the subsection entitled "Managing groups" describes the EMM's Group feature. This can be used to define a group of users or mobile devices, and to associate an Application Management profile to the group. This type of profile defines the applications that are available to devices in the group and the applications that are automatically installed onto devices in the group.

Component Testing Assurance Activities: 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.

The evaluator created an application group, assigned an application to that group, and assigned a user to be a member of that group. The evaluator then used a device belonging to the user and observed that the device could install the application. The evaluator attempted to install the application from a device belonging to a user that was not a member of the group. The evaluator observed the application could not be installed in this case.

2.4.9 Specification of Management Functions (MDMA10:FMT_SMF_EXT.4)

2.4.9.1 MDMA10:FMT_SMF_EXT.4.1

TSS Assurance Activities: None Defined

Guidance Assurance Activities: None Defined

Testing Assurance Activities: None Defined

2.4.9.2 MDMA10:FMT_SMF_EXT.4.2

TSS Assurance Activities: None Defined

Guidance Assurance Activities: None Defined

Testing Assurance Activities: None Defined

Component TSS Assurance Activities: The evaluator shall verify that the any assigned functions are described in the TSS and that these functions are documented as supported by the platform. The evaluator shall examine the TSS to verify that any differences between management functions and policies for each supported mobile device are listed.

The evaluator shall verify that the TSS describes the methods in which the MDM Agent can be enrolled.

The TSS description shall make clear if the MDM Agent supports multiple interfaces for enrollment and configuration (for example, both remote configuration and local configuration).

Section 6.4 of the ST states that enrollment of a device occurs requires an MDM administrator to define a device, user and password on the EMM Server. The EMM Agent on that device is then used (by the mobile device user) to initiate an enrollment action which provides the user's ID and password, along with identification of the device. The EMM Agent component of the TOE is configured with an X.509v3 certificate suitable to facilitate secure communication with the EMM. This certificate is provisioned during device enrollment. The EMM Server can be configured to use an external Microsoft CA to sign CSRs from the EMM Agent during enrollment. Once secure communication is enabled and the device is enrolled, the EMM Agent accepts commands and policies from the enrolled EMM and implements those commands and policies.

Component Guidance Assurance Activities: The evaluator shall verify the AGD guidance includes detailed instructions for configuring each function in this requirement.

If the MDM Agent is a component of the MDM system (i.e. MDM Server is the Base-PP), the evaluator shall verify, by consulting documentation for the claimed mobile device platforms, that the configurable functions listed for this Agent are supported by the platforms.

If the MDM Agent supports multiple interfaces for configuration (for example, both remote configuration and local configuration), the AGD guidance makes clear whether some functions are restricted to certain interfaces.

Section 6 of [Admin] the section entitled "Configuring policies by device platform" describes the various device management profile policies that align with capabilities identified by the FMT_SMF_EXT.4.1 requirement. Section 4 of [Admin] the subsection entitled "List of Device Commands" describes the device commands that can be sent to a mobile device from the MDM server that align with capabilities identified by the FMT_SMF_EXT.4.1 requirement. Refer to the following table for the sections in [Admin] which address each management function.

Management Function	Relevant Section in [Admin]
1. transition to the locked state (MDF Function 6)	**Command:** Lock/Unlock device
2. full wipe of protected data (MDF Function 7)	**Command:** Factory Reset
3. unenroll from management	Section 4 subsection "Changing device status"
4. install policies	**Command:** Apply latest Device/App mgmt. profiles
5. query connectivity status	**Command:** Sync Device Information
6. query the current version of the MD firmware/software	**Command:** Sync Device Information
7. query the current version of the hardware model of the	**Command:** Sync Device Information

device	
8. query the current version of installed mobile applications	**Command:** Sync Device Information
9. import Y.509v3 certificates into the Trust Anchor Database (MDF Function 11)	Section 6 subsection "Configuring Android Legacy Policies"
10. install applications, (MDF Function 16)	Section 6 subsection "Configuring polices by device platform"
11. update system software, (MDF Function 15)	Section 6 subsection "Configuring polices by device platform"
12. remove applications, (MDF Function 14)	Section 6 subsection "Configuring polices by device platform"
13. remove Enterprise applications (MDF Function 17)	Section 6 subsection "Configuring polices by device platform"
14. wipe Enterprise data, (MDF Function 28)	**Command:** Factory Reset
15. remove imported X.509v3 certificates and default X.509v3 certificates in the Trust Anchor Database (MDF Function 12)	Section 6 subsection "Configuring Android Legacy Policies"
17. import keys/secrets into the secure key storage (MDF Function 9)	Section 6 subsection "Configuring Android Legacy Policies"
18. destroy imported keys/secrets and [no other keys/secrets] in the secure key storage (MDF Function 10)	Section 6 subsection "Configuring Android Legacy Policies"
19. read audit logs kept by the MD (MDF Function 32)	**Command:** Collect audit logs
25. password policy: a. minimum password length (MDF Function 1) b. minimum password complexity (MDF Function 1) c. maximum password lifetime (MDF Function 1)	**Policy Values**: Password Policies, minimum strength, Expire after (days), Maximum password length using sequential numbers, Maximum password length sequence of character used
26. session locking policy: a. screen-lock enabled/disabled (MDF Function 2) b. screen lock timeout (MDF Function 2) c. number of authentication failures (MDF Function 2)	**Policy Values**: Device lock timeout (min), Screen lock timeout (sec), Maximum failed attempts, If maximum failed attempts is exceeded
27. wireless networks (SSIDs) to which the MD may connect (WLAN Client EP Function 2)	Section 6 subsection "Configuring Android Legacy Policies"
28. security policy for each wireless network: a. specify the CA(s) from which the MD will accept WLAN authentication server certificate(s) (WLAN Client EP Function 1) b. ability to specify security type (WLAN Client EP Function 1) c. ability to specify authentication protocol (WLAN Client EP Function 1) d. specify the client credentials to be used for authentication (WLAN Client EP Function 1)	Section 6 subsection "Configuring Android Legacy Policies"
29. application installation policy by a. specifying authorized application repository(s) (MDF Function 8)	Section 6 subsection "Configuring polices by device platform"
29. application installation policy by b. specifying a set of allowed applications based on [application name, developer signature] (an application	Section 6 subsection "Configuring polices by device platform"

whitelist) (MDF Function 8) c. denying application installation (MDF Function 8)	
30. enable/disable policy for camera and microphone across device (MDF Function 5)	**Policy Values**: Camera, Microphone
32. enable/disable policy for NFC, Bluetooth, Wi-Fi, and cellular radios (MDF Function 4)	**Policy Values**: Wi-Fi, Bluetooth, Cellular data connection
34. enable/disable policy for protocols supporting remote access (MDF Function 25)	**Policy Value**: Wi-Fi Hotspot, Bluetooth tethering, USB tethering activation
35. enable/disable policy for developer modes (MDF Function 26)	**Policy Value**: USB debugging
36. enable policy for data-at rest protection (MDF Function 20)	Set by default on devices
37. enable policy for removable media's data-at-rest protection (MDF Function 21)	**Policy Value**: Encryption for storage
38. enable/disable policy for local authentication bypass, (MDF Function 27)	**Policy Values:** Block functions setting on lock screen (Trust Agent)
47. the unlock banner policy (MDF Function 36)	**Policy Value**: Reboot banners stationery
48. configure the auditable items (MDF Function 37)	**Policy Value**: Save logs, Log level, Maximum log size (MB), Maximum days for storage (day)
49. enable/disable a. USB mass storage mode (MDF Function 39)	**Policy Value**: PC connection
51. enable/disable a. Hotspot functionality (MDF Function 41) b. USB tethering (MDF Function 41)	**Policy Value**: Wi-Fi Hotspot, Bluetooth tethering, USB tethering activation
52. enable/disable location services: a. across device (MDF Function 22)	**Policy Value**: GPS
55. enable/disable policy for use of Biometric Authentication Factor (MDF Function 23)	**Policy Values:** Block functions setting on lock screen (Fingerprint)

Unenrollment from the management system is handled by deleting the device through the MDM server interface, which is described in Section 4 of [Admin] in the subsection entitled "Deactivating devices”

Component Testing Assurance Activities: Test 1: In conjunction with the evaluation activities in theBase-PP, the evaluator shall attempt to configure each administrator-provided management function and shall verify that the mobile device executes the commands and enforces the policies.

Test 2: The evaluator shall configure the MDM Agent authentication certificate in accordance with the configuration guidance. The evaluator shall verify that the MDM Agent uses this certificate in performing the tests for FPT_ITT.1(2) (MDM as Base-PP) or FTP_ITC_EXT.1(2) (MDF as Base-PP). (TD0491 applied)

Test 3: In conjunction with other evaluation activities, the evaluator shall attempt to enroll the MDM Agent in management with each interface identified in the TSS, and verify that the MDM Agent can manage the device and communicate with the MDM Server.

Test 4: [conditional] In conjunction with the evaluation activity for FAU_ALT_EXT.2.1, the evaluator shall configure the periodicity for reachability events for several configured time periods and shall verify that the MDM Server receives alerts on that schedule.

Test 5: [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 mobile device.

Test 1: The testing performed for FMT_SMF.1(1) address all of the required commands and management functions. Within that test case, not only are actions taken to demonstrate that the MDM server can perform the commands and management function, but steps within the test case also demonstrate that the commands and management function results in the expected change to the configuration of the mobile device.

Test 2: These activities can be found as part of the testing of FPT_ITT.1(1), in which an EMM Agent enrolled and performed various interactions with the EMM. For each interaction, the EMM agent and EMM used TLS, encrypted the traffic, and authenticated the peer using certificates.

Test 3: These activities can be found as part of the testing of FPT_ITT.1(1), in which an EMM Agent enrolled and performed various interactions with the EMM. For each interaction, the EMM agent and EMM used TLS, encrypted the traffic, and authenticated the peer using certificates.

Test 4: The evaluator configured 4, 5, and 6 hour time period, and collected audit events showing the reachability events occurred as scheduled.

Test 5: The testing performed for FMT_SMF.1(1) address importing certificates as well as controlling the configuration and policies applied to mobile devices.

2.4.10 Security Management Roles (MDMPP40:FMT_SMR.1(1))

2.4.10.1 MDMPP40:FMT_SMR.1.1(1)

TSS Assurance Activities: None Defined

Guidance Assurance Activities: None Defined

Testing Assurance Activities: None Defined

2.4.10.2 MDMPP40:FMT_SMR.1.2(1)

TSS Assurance Activities: None Defined

Guidance Assurance Activities: None Defined

Testing Assurance Activities: None Defined

Component TSS Assurance Activities: The evaluator shall examine the TSS to verify that it describes the administrator role and the powers granted to and limitations of the role.

Section6.4 of the ST states the EMM Server provides several different roles: server primary administrators, security configuration administrators, device user administrators, auditor, and MD users. Server primary administrators are administrators that have an administrative account on the underlying Microsoft Windows Server platform (i.e., the Windows administrator), log into Windows locally or through RDP, and are responsible for installation, install configuration, and monitoring of Windows/platform level audit logs. Security configuration administrators log into the EMM Server's HTTPS WebUI and are responsible for configuring the EMM's settings. Device user administrators also login through the EMM Server's HTTPS WebUI and are responsible for setting up accounts for mobile device users, inspecting the status of a given mobile device, and revoking/unenrolling a mobile device. Finally, auditors (who also login through the EMM Server's HTTPS WebUI) have permissions only to access the EMM Console's audit log. All Administrators (other than the server primary administrator) connect remotely to the Server via HTTPS (using a standard web browser) and must be authentication (providing a username and password) before gaining any access to the Server. The Server requires that administrator accounts be created for each administrator, and separates such administrators from MD users (unless an administrator has explicitly created a separate administrative account for the user). The Server allows MD users to enroll their mobile devices and thus allows MD users to have the EMM manage their mobile devices to secure organization data and access.

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

The TOE only supports administration of the product through the HTTPS/TLS web interface, and all documentation for the administrator describes use of this interface.

Component Testing Assurance Activities: 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.

The evaluator connected to the EMM Server as an administrator using only the remote interface (HTTPS). The evaluator performed a function (e.g., the steps in FTA_TAB.1 Test 1)..

2.4.11 Security Management Roles (MAS Server) (MDMPP40:FMT_SMR.1(2))

2.4.11.1 MDMPP40:FMT_SMR.1.1(2)

TSS Assurance Activities: None Defined

Guidance Assurance Activities: None Defined

Testing Assurance Activities: None Defined

2.4.11.2 MDMPP40:FMT_SMR.1.2(2)

TSS Assurance Activities: None Defined

Guidance Assurance Activities: None Defined

Testing Assurance Activities: None Defined

Component TSS Assurance Activities: The evaluator shall examine the TSS to verify that it describes the administrator role and the powers granted to and limitations of the role.

The MAS and EMM server are the same. See MDMPP40:FMT_SMR.1.2(1) for a description of roles.

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

Since the MAS server and the MDM server are the same entity, there are no additional interfaces offered by the MAS server which are not part of the MDM server.

Component Testing Assurance Activities: 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.

Because the MAS server and the MDM server are the same software entity, no further testing was performed beyond that described for FMT_SMR.1(1)-t1.

2.4.12 User Unenrollment Prevention (MDMA10:FMT_UNR_EXT.1)

2.4.12.1 MDMA10:FMT_UNR_EXT.1.1

TSS Assurance Activities: None Defined

Guidance Assurance Activities: None Defined

Testing Assurance Activities: None Defined

Component TSS Assurance Activities: The evaluator shall ensure that the TSS describes the mechanism used to prevent users from unenrolling or the remediation actions applied when unenrolled.

Section 6.4 of the ST states the administrator can configure (using the EMM Console) the EMM Agent to prevent the mobile phone's user from removing the Agent's administrative privileges, thus preventing the user from unenrolling the Agent. If an administrator has not restricted the mobile phone user's ability to remove the EMM Agent's administrative privileges, then the user can remove the EMM Agent's administrative privileges (unenrolling it from the EMM). Finally, the administrator can forcibly unenroll the EMM Agent from EMM (and if the Agent will receive the unenrollment request when it has network connectivity to the EMM).

Component Guidance Assurance Activities: The evaluator shall ensure that the administrative guidance instructs administrators in configuring the unenrollment prevention in each available configuration interface. If any configuration allows users to unenroll, the guidance also describes the actions that unenroll the Agent.

Section 4 of [Admin] in the subsection entitled "Deactivating devices" explains how an administrator can unenroll (deactivate or delete) a mobile device using the MDM server interface. An administrator can configure the mobile devices to prevent deletion of the EMM applications using the instructions in section 5 of [Admin], under the heading "Preventing EMM applications being forged or deleted". This material defines how to set the EMM applications as mandatory. Additionally, configuring the policies such that users are allowed or disallowed to perform un-enrollment can be accomplished by using the "Setting EMM client policies" setting described in section 11 of [Admin]. The image in this section shows the "Applicable policies for EMM Client" screen used to set the un-enrollment policy via a checkbox labelled "Allow Unenroll request.".

Component Testing Assurance Activities: Test 1: If 'prevent the unenrollment from occurring' is selected: The evaluator shall configure the Agent according to the administrative guidance for each available configuration interface, shall attempt to unenroll the device, and shall verify that the attempt fails.

Test 2: If 'apply remediation actions' is selected: If any configuration allows the user to unenroll, the evaluator shall configure the Agent to allow user unenrollment, attempt to unenroll, and verify that the remediation actions are applied.

Test 1: The evaluator configured a policy to prevent un-enrollment of a mobile device, ensured that the policy was applied to the MD, and attempted steps equivalent to un-enrollment (i.e., EMM agent uninstall). The evaluator observed that the un-enrollment was not successful.

Test 2: The ST does not make the selection for this conditional test.

2.5 Protection of the TSF (FPT)

2.5.1 Use of Supported Services and APIs (MDMPP40:FPT_API_EXT.1)

2.5.1.1 MDMPP40:FPT_API_EXT.1.1

TSS Assurance Activities: None Defined

Guidance Assurance Activities: None Defined

Testing Assurance Activities: None Defined

Component TSS Assurance Activities: 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.

Appendix A provides the API used by the TOE. The evaluator searched the java and SQL web pages and found the interfaces do exist.

Component Guidance Assurance Activities: None Defined

Component Testing Assurance Activities: None Defined

2.5.2 Internal TOE TSF Data Transfer (MDMPP40:FPT_ITT.1(1))

2.5.2.1 MDMPP40:FPT_ITT.1.1(1)

TSS Assurance Activities: None Defined

Guidance Assurance Activities: None Defined

Testing Assurance Activities: None Defined

Component TSS Assurance Activities: 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).

Section 6.5 of the ST states the EMM (i.e., EMM Server, Push Server, AppTunnel Server, Push Proxy and AT-Relay) utilize TLS (with mutual/client authentication using configured X509 certificates) as the trusted channel to protect all data transmitted among its distributed parts including the EMM server components and EMM Agents from disclosure and modification. The EMM Agent utilizes the TLS functions in its platform device for TLS and X509 to protect the communication channels between itself and the EMM server components via TLS mutually authenticated using X509 certificates. The use of TLS by the TOE as both client and server is consistent with the requirements for protocols claimed in the ST.

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

Section 1 of [Admin] in the subsection entitled "Common Criteria Evaluated Security Functions". This section indicates that communication between the MDM agent and MDM server is secured using TLS channels by default. This section also presents diagrams depicting the protections provided for various communication channels between the EMM Server, App Tunnel server, and push server. All of these channels are protected using TLS when the installation instructions contained by [Install] are followed.

Component Testing Assurance Activities: Test 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 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.

Test 1 & 2: The evaluator started a packet capture to collect all traffic between the distributed EMM server components. The evaluator then restarted the entire MDM system and let it operate for several minutes until all applicable connections could be observed in the packet capture. The evaluator reviewed the packet capture and determined that all network traffic between distributed EMM server components was protected by TLS as claimed.

2.5.3 Internal TOE TSF Data Transfer (MDM Agent) (MDMPP40:FPT_ITT.1(2))

2.5.3.1 MDMPP40:FPT_ITT.1.1(2)

TSS Assurance Activities: None Defined

Guidance Assurance Activities: None Defined

Testing Assurance Activities: None Defined

Component TSS Assurance Activities: 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).

Section 6.5 of the ST states the EMM (i.e., EMM Server, Push Server, AppTunnel Server, Push Proxy and AT-Relay) utilize TLS (with mutual/client authentication using configured X509 certificates) as the trusted channel to protect all data transmitted among its distributed parts including the EMM server components and EMM Agents from disclosure and modification. The EMM Agent utilizes the TLS functions in its platform device for TLS and X509 to protect the communication channels between itself and the EMM server components via TLS mutually authenticated using X509 certificates. The use of TLS by the TOE as both client and server is consistent with the requirements for protocols claimed in the ST.

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

Section 1 of [Admin] in the subsection entitled "Common Criteria Evaluated Security Functions." This section indicates that communication between the MDM agent and MDM server is secured using TLS channels by default. This section also presents diagrams depicting the protections provided for various communication channels between the EMM Server, App Tunnel server, and push server. All of these channels are protected using TLS. No special configuration is required beyond the configuration described throughout the normal install instructions in [Install]..

Component Testing Assurance Activities: Test 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 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.

Test 1 & 2:The evaluator started a packet capture to collect traffic between a mobile device (using TLS and X509 implemented in the platform device) and the EMM server components (using TLS directly implemented in the TOE). The evaluator then used the EMM agent to enroll the mobile device, push policies, download applications,

and issue various commands. The evaluator reviewed the packet capture and determined that all network traffic associated with these actions was protected by TLS.

2.5.4 Use of Third Party Libraries (MDMPP40:FPT_LIB_EXT.1)

2.5.4.1 MDMPP40:FPT_LIB_EXT.1.1

TSS Assurance Activities: None Defined

Guidance Assurance Activities: None Defined

Testing Assurance Activities: None Defined

Component TSS Assurance Activities: The evaluator shall verify that the TSS lists the libraries used by the MDM software. The evaluator shall verify that libraries found to be packaged with or employed by the MDM software are limited to those in the assignment.

Section 6.5 of the ST states the EMM links with the following 3rd party libraries:

- RSA BSAFE Cryto-J,
- Ext JS / Ext JS-org.webjars:extjs,

and the EMM Agent links with the following 3rd party library:

- Eldos Coporation Solid File System

Component Guidance Assurance Activities: None Defined

Component Testing Assurance Activities: None Defined

2.5.5 Functionality Testing (MDMPP40:FPT_TST_EXT.1)

2.5.5.1 MDMPP40:FPT_TST_EXT.1.1

TSS Assurance Activities: None Defined

Guidance Assurance Activities: None Defined

Testing Assurance Activities: None Defined

2.5.5.2 MDMPP40:FPT_TST_EXT.1.2

TSS Assurance Activities: None Defined

Guidance Assurance Activities: None Defined

Testing Assurance Activities: None Defined

Component TSS Assurance Activities: 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.

Section 6.5 of the ST states the EMM performs power-up tests to ensure correct operation. The EMM's Crypto-J library performs power-up Known Answer Tests for each of its cryptographic algorithms (including AES, RSA, ECDSA, SHA, HMAC-SHA) to ensure correct operations, and the EMM as a whole performs a startup integrity check of its executable code to ensure that the calculated SHA-256 hash of the code during startup matches its expected value.. Should the startup integrity check fail, the server component will log the error and attempt to continue loading. This ensures that corrupted TOE code is detected.

The EMM Agent relies upon its platform to perform a test of its cryptographic algorithms upon power-up.

Component Guidance Assurance Activities: None Defined

Component Testing Assurance Activities: The evaluator shall perform the following tests:

Test 1: The evaluator performs the integrity check on a known good TSF executable and verifies that the check is successful.

Test 2: The evaluator modifies the TSF executable, performs the integrity check on the modified TSF executable and verifies that the check fails.

Tet 1: The EMM performs integrity checks upon boot. Thus, the evaluator booted a known good version of the installed system and monitored the Audit logs. The audit log contained messages that indicated that integrity checks for TSF executables were successful.

Test 2: The evaluator then modified a TSF binary and restarted the EMM. The result was that the file corruption was detected and the EMM failed to start. The EMM log also indicated the file which failed its integrity check.

2.5.6 Trusted Update (MDMPP40:FPT_TUD_EXT.1)

2.5.6.1 MDMPP40:FPT_TUD_EXT.1.1

TSS Assurance Activities: None Defined

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

The section entitled "Viewing EMM application version", in section 5 of [Admin] provides instructions on how to determine the version of the EMM Agent software on the EMM Server that can be pushed out to mobile devices. The section entitled "Viewing the EMM server information", in section 11 of [Admin] provides instructions to information specific to the EMM including the EMM version.

Testing Assurance Activities: The evaluator shall query the TSF for the current version of the software according to the AGD guidance and shall verify that the current version matches that of the documented and installed version.

The evaluator followed the instructions for “Viewing EMM application version” and found that the queried version matched the CC evaluated version.

2.5.6.2 MDMPP40:FPT_TUD_EXT.1.2

TSS Assurance Activities: 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).

Section 6.5 of ST indicates that an administrator must obtain updates and explicitly update the TOE using that update by using operations provided by the platform user interface. The EMM’s platform checks the Microsoft AuthentiCode signature embedded in the update file.

Guidance Assurance Activities: None Defined

Testing Assurance Activities: None Defined

2.5.6.3 MDMPP40:FPT_TUD_EXT.1.3

TSS Assurance Activities: 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.

Section 6.5 of ST indicates that an administrator must obtain updates and explicitly update the TOE using that update. The EMM's platform checks the Microsoft AuthentiCode signature embedded in the update file.

Guidance Assurance Activities: The evaluator shall examine the AGD 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.

The section entitled "Viewing the EMM server information", in section 11 of [ADMIN] describes the server information that is available about the current system configuration. This includes the EMM version. Section 5 of [INSTALL] entitled "Updating EMM" describes how to update the TOE including how to start an update and how to verify the digital signature of an update (Section 5.2.1 "Checking digital signature").

Testing Assurance Activities: The evaluator shall perform the following tests:

Test 1: The evaluator shall attempt to initiate an update digitally signed by the vendor and verify that the update is successfully installed.

Test 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.

Test 1: The evaluator followed guidance to initiate a TOE update referring to a properly signed update provided by the vendor. The update changed the TOE version and verified the update RPMs.

Test 2: The evaluator followed guidance to initiate a TOE update referring to an unsigned update provided by the vendor. The update attempt detected that the new RPMs were not properly signed and did not perform the update action.

Component TSS Assurance Activities: None Defined

Component Guidance Assurance Activities: None Defined

Component Testing Assurance Activities: None Defined

2.6 TOE ACCESS (FTA)

2.6.1 DEFAULT TOE ACCESS BANNERS (MDMPP40:FTA_TAB.1)

2.6.1.1 MDMPP40:FTA_TAB.1.1

TSS Assurance Activities: None Defined

Guidance Assurance Activities: None Defined

Testing Assurance Activities: None Defined

Component TSS Assurance Activities: 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.

Section 6.6 of the ST states an Administrator can configure the EMM Server to display an Administrator-specified advisory notice and consent warning message (in the form of a copyright message) regarding use of the EMM Server. The logo and login notification are shown on the EMM Server login page.

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

The section entitled "Configuring the environment", in section 2 of [Admin] explains how to set various server configuration values, including the 'Copyright' information. This information is displayed on the login page.

Component Testing Assurance Activities: 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.

The evaluator followed guidance to specify TOE banner. The evaluator observed that the banner configured appeared in the login window for the EMM Server's web console.

2.7 TRUSTED PATH/CHANNELS (FTP)

2.7.1 Inter-TSF Trusted Channel (Authorized IT Entities) (MDMPP40:FTP_ITC.1(1))

2.7.1.1 MDMPP40:FTP_ITC.1.1(1)

TSS Assurance Activities: 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).

Section 6.7 of the ST states the EMM uses communication channels to an MDM Agent that is are internal to the TOE (i.e., the Android agent), external to the TOE (i.e., the Apple agent), as well as communication channels to other parts of the distributed TOE. The EMM uses TLS to secure communication with EMM Agents and HTTPS to secure communication with administrators (through their browser) where audit records can be exported. Since the EMM provides the MAS server functionality, there are no additional communication paths for the MAS audit communications. The EMM communicates with the Apple agent using TLS to secure the communication pathway with either initiating the communication. The EMM depends on a database server and Microsoft Certificate Authority and it utilizes IPsec provided by its host operating system to secure those communication channels when the syslog server, database server and/or Certificate Authority are not operating on the same host

Guidance Assurance Activities: None Defined

Testing Assurance Activities: None Defined

2.7.1.2 MDMPP40:FTP_ITC.1.2(1)

TSS Assurance Activities: 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).

Section 6.7 of [ST] explains that the EMM depends on a database server and the EMM utilizes IPsec provided by its host operating system to secure that communication channel when the database server is not operating on the same host and the syslog server. This section also explains that the EMM uses TLS to secure communication with EMM Agents and HTTPS to secure communication with administrators (through their browser) where audit records can be exported.

Guidance Assurance Activities: None Defined

Testing Assurance Activities: None Defined

2.7.1.3 MDMPP40:FTP_ITC.1.3(1)

TSS Assurance Activities: 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).

Section 6.7 of [ST] explains that the EMM depends on a database server and the EMM utilizes IPsec provided by its host operating system to secure that communication channel when the database server is not operating on the same host and the syslog server. This section also explains that the EMM uses TLS to secure communication with EMM Agents and HTTPS to secure communication with administrators (through their browser) where audit records can be exported

Guidance Assurance Activities: 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.

The MDM server provides a trusted channel for export of audit data. Section 12 of [Admin], subsection entitled “Exporting audit logs to an Excel file” explains that to export audit data from the EMM logs, the administrator must use the "Export" button on the Audit logs screens. The guidance explains that the EMM audit log data is exported to the user's computer through the HTTPS/TLS or IPsec protected connection established when the administrator logged into the EMM. The guidance in Section 13 of [Admin] the subsection entitled "List of Audit Events" explains that audit records contained by these files are not exported through the EMM console and are not available for export through syslog. These audit records must be exported using a secured RDP connection to the Windows server platform on which the EMM, Push and/or AT server are running, then export the audit data through that RDP connection.

The subsection entitled “Common Criteria Evaluated Security Functions” in Section 1 of [Admin] indicates that the use of TLS to protect the remote administration communication channel is the default behavior. Additionally, Appendix B.2 of [Install] “Setting Tomcat” describes how to configure the ciphersuites used by the EMM Admin portal (aka the Web UI).

The subsection entitled “Common Criteria Evaluated Security Functions” in Section 1 of [Admin] also indicates that the communication channels to the external CA and SQL servers are protected by the platform using IPsec as described by [IPsec]. Instructions to configure IPsec are provided in [IPsec]. The subsection entitled “Certificate authority (CA)” in section 9 of [Admin] describes how to configure the CA certificates used to protect communications between the TOE and the external CA. Appendix F “Installing SQL Server Certificate” provides instructions to configure the certificate used to protect communications between the TOE and the external SQL

server. Other configuration data for the SQL server is set during TOE installation which is described in section 3 of [Install].**Testing Assurance Activities**: Test 1: The evaluators 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 2: The evaluator shall ensure, for each method of communication, the channel data is not sent in plaintext.

Test 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.

For this TOE, the export of audit records is provided through the HTTPS/TLS admin console connection where audit can be downloaded in the form of Excel files. Similarly, device information can be exported using the same HTTPS/TLS console interface. See Test Case FTP_TRP.1(1)-t1 where a packet capture shows that the console session (including audit and device information export) is fully encapsulated using TLSv1.2.

Finally, a packet capture of database traffic protected by IPSec was examined and found that data between the TOE and external services including the database, syslog server, and Certificate Authority Server was not plaintext and was protected by IPsec.

Component TSS Assurance Activities: None Defined

Component Guidance Assurance Activities: None Defined

Component Testing Assurance Activities: None Defined

2.7.2 Inter-TSF Trusted Channel (MDM Agent) (MDMPP40:FTP_ITC.1(2))

2.7.2.1 MDMPP40:FTP_ITC.1.1(2)

TSS Assurance Activities: 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).

Section 6.7 of the ST states the EMM uses TLS to secure communication with EMM Agents. The TLS protocol is covered by the already included TLS requirements.

Guidance Assurance Activities: None Defined

Testing Assurance Activities: None Defined

2.7.2.2 MDMPP40:FTP_ITC.1.2(2)

TSS Assurance Activities: 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).

Section 6.7 of the ST states the EMM uses TLS to secure communication with EMM Agents. The TLS protocol is covered by the already included TLS requirements.

Guidance Assurance Activities: None Defined

Testing Assurance Activities: None Defined

2.7.2.3 MDMPP40:FTP_ITC.1.3(2)

TSS Assurance Activities: 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).

Section 6.7 of the ST states the EMM uses TLS to secure communication with EMM Agents. The TLS protocol is covered by the already included TLS requirements.

Guidance Assurance Activities: 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.

The communication between the EMM agent and the EMM utilize TLS configured using default values meeting FCS_TLSC_EXT.1 (i.e., using only allowed protocols and ciphersuites). No evaluation specific administrative configuration is required to meet the requirements.

Testing Assurance Activities: Test 1: The evaluators 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 2: The evaluator shall ensure, for each method of Agent-Server communication, the channel data is not sent in plaintext.

Test 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.

For this requirement the evaluator exercised the connection between the EMM and the iOS agent. Communications between the EMM and the EMM Android agent are addressed by FPT_ITT.1(1).

Test 1: The TLS protocol is thoroughly tested as part of the corresponding protocol requirements (e.g., FCS_TLSS_EXT.1). For each IT entity a connection using each supported protocol is established to demonstrate that protected communication using that protocol is supported. Packet captures were obtained for the initiation and use of the communication related to communication with an iOS agent for the purpose of enrollment, pushing policies, downloading applications, and other management functions.

Test 2: The packet captures obtained during test 1 were reviewed and the evaluator observed that the network communication was not in plaintext.

Test 3: The initiation of the connection for communication related to communication with an iOS agent were captured and revealed that TLSv1.2 was being used.

Component TSS Assurance Activities: None Defined

Component Guidance Assurance Activities: None Defined

Component Testing Assurance Activities: None Defined

2.7.3 Trusted Channel (MDMPP40:FTP_ITC_EXT.1)

2.7.3.1 MDMPP40:FTP_ITC_EXT.1.1

TSS Assurance Activities: None Defined

Guidance Assurance Activities: None Defined

Testing Assurance Activities: None Defined

Component TSS Assurance Activities: The evaluator shall ensure that the TSS contains whether the MDM Server communication channel is internal or external to the TOE.

Section 6.7 of ST explains that the EMM uses communication channels to an MDM Agent that is are internal to the TOE (i.e., the Android agent), external to the TOE (i.e., the Apple agent), as well as communication channels to other parts of the distributed TOE.

Component Guidance Assurance Activities: None Defined

Component Testing Assurance Activities: This testing can be completed in conjunction with the testing for FPT_ITT.1(1)/FPT_ITT.1(2), FTP_ITC.1(2) or FTP_ITC.1(3).

This testing was completed in conjunction with the testing for FPT_ITT.1(1)/FPT_ITT.1(2), FTP_ITC.1(1) or FTP_ITC.1(2).

2.7.4 Trusted Path (for Remote Administration) (MDMPP40:FTP_TRP.1(1))

2.7.4.1 MDMPP40:FTP_TRP.1.1(1)

TSS Assurance Activities: 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).

As described by section 6.7 of ST, the EMM Server implements a web-based user interface for remote administration. This web-based interface is accessible through HTTPS/TLS. Because the TOE includes an HTTPS server-side interface, the FCS_HTTPS_EXT.1 and FCS_TLSS_EXT.1 requirements are included in ST.

Guidance Assurance Activities: None Defined

Testing Assurance Activities: None Defined

2.7.4.2 MDMPP40:FTP_TRP.1.2(1)

TSS Assurance Activities: 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).

As described by section 6.7 of ST, the EMM Server implements a web-based user interface for remote administration. This web-based interface is accessible through HTTPS/TLS. Because the TOE includes an HTTPS server-side interface, the FCS_HTTPS_EXT.1 and FCS_TLSS_EXT.1 requirements are included in ST.

Guidance Assurance Activities: None Defined

Testing Assurance Activities: None Defined

2.7.4.3 MDMPP40:FTP_TRP.1.3(1)

TSS Assurance Activities: 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).

As described by section 6.7 of ST, the EMM Server implements a web-based user interface for remote administration. This web-based interface is accessible through HTTPS/TLS. Because the TOE includes an HTTPS server-side interface, the FCS_HTTPS_EXT.1 and FCS_TLSS_EXT.1 requirements are included in ST..

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

Login to the server occurs exclusively using a TLS/HTTPS communication path from the administrator's web browser to the EMM Server. The use of TLS/HTTPS is the default behavior and no configuration is necessary to protect communication between a browser and the EMM Server. The subsection entitled "Common Criteria Evaluated Security Functions" in Section 1 of [Admin] indicates that the use of TLS to protect this communication channel is the default behavior. Instructions for login can be found in the subsection entitled "Signing in to EMM" in Section 2 of [Admin]. Other configuration data for the EMM is set during TOE installation which is described in section 3 of [Install].

Testing Assurance Activities: The evaluator shall also perform the following tests:

Test 1: The evaluators 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 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 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.

Test 1: The evaluator connected to the EMM Server as an administrator using all claimed remote interface (i.e., TLS). Once connected the evaluator performed at least one management activity using that interface. The evaluator collected the network traffic associated with this action that was transmitted between the MDM server and remote administrator (using a sniffer). The evaluator examined the collected network traffic and observed that the traffic was protected from disclosure and modification by TLS v1.2 (and that traffic was encrypted).

Test 2: As a result of a review of guidance documentation, the evaluator identified no path for administration other than the web-based administrative portal and access to the host servers (e.g., via RDP or direct console access). Note that the host platform Windows Server 2012 R2 has been evaluated and is on the NIAP PCL (https://www.niap-ccevs.org/Product/PCL.cfm).

Test 3: Review of the packet capture from test 1 above, showed that the traffic between the EMM Server and the remote administrator was not plaintext.

Component TSS Assurance Activities: None Defined

Component Guidance Assurance Activities: None Defined

Component Testing Assurance Activities: None Defined

2.7.5 Trusted Path (for Enrollment) (MDMPP40:FTP_TRP.1(2))

2.7.5.1 MDMPP40:FTP_TRP.1.1(2)

TSS Assurance Activities: 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).

Section 6.7 of ST indicates that the TOE implements a HTTPS/TLS connection with the EMM agent for enrollment. Because the TOE includes an HTTPS server side interface, the FCS_TLSS_EXT.1 requirement is included in ST.

Guidance Assurance Activities: None Defined

Testing Assurance Activities: None Defined

2.7.5.2 MDMPP40:FTP_TRP.1.2(2)

TSS Assurance Activities: 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).

Section 6.7 of ST indicates that the TOE implements a HTTPS/TLS connection with the EMM agent for enrollment. Because the TOE includes an HTTPS server side interface, the FCS_TLSS_EXT.1 requirement is included in ST.

Guidance Assurance Activities: None Defined

Testing Assurance Activities: None Defined

2.7.5.3 MDMPP40:FTP_TRP.1.3(2)

TSS Assurance Activities: 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).

Section 6.7 of ST indicates that the TOE implements a HTTPS/TLS connection with the EMM agent for enrollment. Because the TOE includes an HTTPS server side interface, the FCS_TLSS_EXT.1 requirement is included in ST.

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

Section 6.3 of ST indicates that the only means for enrollment occurs from a MD by establishing an HTTPS connection to the MDM server. This is referred to as "Activating" a device. Section 4 of [Admin] the subsection entitled "Activating devices" describes the process by which the user must present a username, mobile device ID, and password to the EMM Agent. Once this information has been provided, the EMM agent establishes an HTTPS connection to the EMM. The subsection entitled "Common Criteria Evaluated Security Functions" in Section 1 of [Admin] indicates that all communications between the web browser and the EMM is encrypted using TLS/HTTPS.

Testing Assurance Activities: For each MDM Agent/platform listed as supported in the ST:

Test 1: The evaluators 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 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 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.

Test 1-3: The communication actions performed between an agent (both EMM Android and iOS) and the EMM for enrollment are demonstrated by test MDMPP40:FTP_TRP.1.3(2)-t1. In those test cases the packet captures show:

- the use of TLS for enrollment for both Android and iOS,
- that the TLS version is version 1.2, and
- that data transmitted through TLS for enrollment is not plaintext.

Test 2 & 3: The evaluator followed guidance and found no obvious ways to enroll outside of TLS. Test case MDMPP40:FTP_TRP.1.3(2)-t2 shows the enrollment interfaces presented by the client apps and concludes there was no alternate method of enrollment. The evaluator found that the packet capture of the enrollment showed the enrollment was TLS and was not plaintext.

Component TSS Assurance Activities: None Defined

Component Guidance Assurance Activities: None Defined

Component Testing Assurance Activities: None Defined

3. Protection Profile SAR Assurance Activities

3.1 Development (ADV)

3.1.1 Basic Functional Specification (ADV_FSP.1)

Assurance Activities: 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.

Since functional specification documentation is provided to support the evaluation activities described in the claimed Protection Profiles (as well as packages and/or modules), the successful completion of assurance activities associated with those evaluation activities implies an adequate FSP has been provided.

3.2 Guidance documents (AGD)

3.2.1 Operational User Guidance (AGD_OPE.1)

Assurance Activities: 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/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.

The [Admin] documents provide information to allow device users and administrators to operate the TOE features properly. The additional information specific to this assurance activity is identified by the following list.

a) [Install] and [Admin] contain information explaining how to load certificates, configure TOE-to-TOE communication channels. [Install] does explain how the TLS ciphersuites used by the EMM, Push and AppTunnel servers can be configured. However, no configuration beyond the default is necessary to meet the FCS_TLSC_EXT.1 requirement.

b) Section 5 of [Install] explains the steps necessary to perform an update of the EMM.

c) Section 1.2 of [Install] describes the architecture of the TOE and refers the reader to the Security Target ST for a description of evaluated security features.

3.2.2 Preparative Procedures (AGD_PRE.1)

Assurance Activities: 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 theTOE in the ST.

Detailed step-by-step instructions for installing the TOE are provided by [Install]. There are no special configuration needs associated with the variety of TSF platforms included in the evaluation.

3.3 Life-cycle support (ALC)

3.3.1 Labelling of the TOE (ALC_CMC.1)

Assurance Activities: The evaluator shall check the ST to ensure that it contains an identifier (such as a product name/version number) that specifically identifies the version that meets the requirements of the ST. Further, the evaluator shall check the AGD guidance and TOE samples received for testing to ensure that the version number is consistent with that in the ST. If the vendor maintains a website advertising theTOE, the evaluator shall examine the information on the website to ensure that the information in the ST is sufficient to distinguish the product.

The evaluator verified that the ST, TOE and Guidance are all labelled with the same software name and version. The information is specific enough to procure the TOE and it includes hardware models and software versions. The evaluator checked the TOE software version and hardware identifiers during testing by examining the actual machines that were used for testing.

3.3.2 TOE CM Coverage (ALC_CMS.1)

Assurance Activities: The 'evaluation evidence required by the SARs' in this PP is limited to the information in the ST coupled with the guidance provided to administrators and users under the AGD requirements. By ensuring that the TOE is specifically identified and that this identification is consistent in the ST and in the AGD guidance (as done in the evaluation activity for ALC_CMC.1), the evaluator implicitly confirms the information required by this component.

Life-cycle support is targeted aspects of the developer's life-cycle and instructions to providers of applications for the developer's devices, rather than an in-depth examination of the TSF manufacturer's development and configuration management process. This is not meant to diminish the critical role that a developer's practices play in contributing to the overall trustworthiness of a product; rather, it's a reflection on the information to be made available for evaluation.

The evaluator shall ensure that the 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 theTSF using this unique identification.

See 3.3.1 for an explanation of how all CM items are addressed.

The TSF is uniquely identified as Samsung SDS Co., LTD EMM version 2.2.5.

3.4 Tests (ATE)

3.4.1 Independent Testing - Conformance (ATE_IND.1)

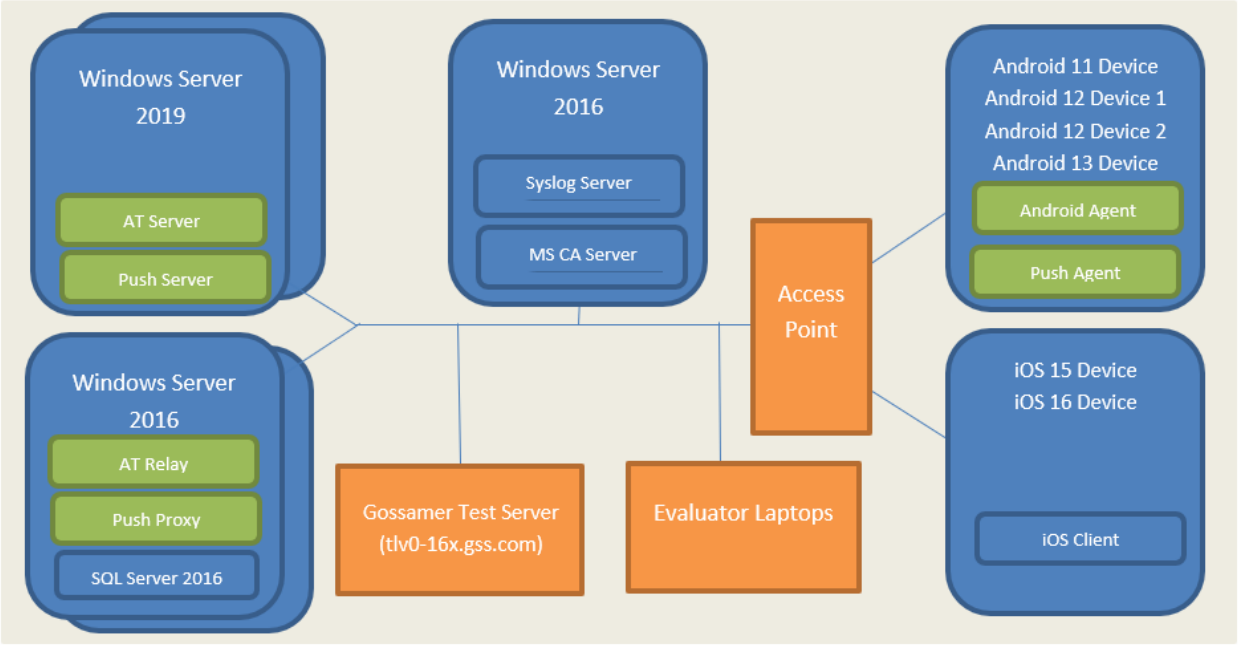
Assurance Activities: The evaluator shall prepare a test plan and report documenting the testing aspects of the system. The test plan covers all of the testing actions contained in the CEM and the body of this PP's Evaluation Activities. While it is not necessary to have one test case per test listed 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.

The evaluator created a proprietary Detailed Test Report (DTR) to address all aspects of this requirement. The DTR discusses the test configuration, test cases, expected results, and test results. The following diagram shows the test environment.



The following TOE components were tested:

- EMM Components: Version 2.2.5
- Agent Applications:
  - Android
    - Client (Agent) Version 2.2.5
    - Push Agent: Version 2.2.5
  - iOS
    - Client Version 2.2.5

The evaluator used the following test tools: Windows, Linux, Putty, Wireshark, Freeradius, OpenSSL, web server, adb, tcpdump, micro-httpd, and Gossamer developed test programs

3.5 Vulnerability assessment (AVA)

3.5.1 Vulnerability Survey (AVA_VAN.1)

Assurance Activities: 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.

The vulnerability analysis is in the proprietary Detailed Test Report (DTR) prepared by the evaluator. The vulnerability analysis includes a public search for vulnerabilities. No unaddressed CVE bulletins were discovered. The evaluator searched the National Vulnerability Database (https://web.nvd.nist.gov/view/vuln/search) and Vulnerability Notes Database (http://www.kb.cert.org/vuls/) on 5/30/2023 with the following search terms: "Crypto-J", "CryptoJ", "Crypto J", "SDS", "Enterprise Mobility Management", "EMM".