Samsung Android 7 on Galaxy Devices

Guidance documentation

Version 3.0

June 1, 2017

Document management

Document identification

Document ID	Samsung Guidance documentation 3.0
Document title	Samsung Android 7 on Galaxy Devices Guidance documentation
Release authority	

Document history

Version	Date	Description	Author
0.1	16-October-2013	Initial draft	
0.2	20-December-2013	Initial draft for Samsung review.	
0.5	January 31, 2014	Update for Android 4.4	Brian Wood
0.6	February 3, 2014	Updated CC Mode API	Brian Wood
0.7	February 10, 2014	Updated based on feedback from CC evaluator	Brian Wood
0.8	February 11, 2014	Added info about determining versions of device, OS & apps	Brian Wood
0.9	February 12, 2014	Added info about obtaining API SDK	Brian Wood
0.10	February 13, 2014	Updates CC Mode app settings	Brian Wood
0.11	February 20, 2014	Added versioning information	Ed Morris
1.1	March 31, 2014	Updated for Galaxy S5/Note 10.1	Brian Wood
1.2	April 3, 2014	Added CRL Checking to the list of required settings	Brian Wood
1.3	April 23, 2014	Corrected Max Password value range	Brian Wood
1.4	April 29, 2014	Updated to show VPN release number	Brian Wood
1.5	April 30, 2014	Removed device locking on password failure	Brian Wood

Version	Date	Description	Author
1.5a	May 2, 2014	Updated device list	Brian Wood
1.5b	June 6, 2014	Modified device list table	Brian Wood
1.6	August 1, 2014	Updated for new devices and options	Brian Wood
1.7	September 7, 2014	Updated for KNOX configurations & devices	Brian Wood Sung Whan Moon
1.8	September 15, 2014	Edited container disable list	Brian Wood
1.9	September 18, 2014	Edits based on KNOX eval feedback	Brian Wood
1.10	September 19, 2014	Updated device list	Brian Wood
1.11	October 7, 2014	Edited versions and CC Mode access	Brian Wood
1.12	October 20, 2014	Edited CC mode access	Brian Wood
1.13	October 28, 2014	Updated device list	Brian Wood
1.14	October 30, 2014	Edits based on Validator feedback	Brian Wood
2.0	December 18, 2014	Edited for Android 5	Brian Wood
2.1	April 9, 2015	Updated device list	Brian Wood
2.2	July 31, 2015	Updated device list	Brian Wood
2.3	October 1, 2015	Updated device list	Brian Wood
2.4	April 19, 2016	Updated device list & features	Brian Wood
2.5	October 4, 2016	Updated device list & features	Brian Wood
3.0	June 1, 2017	Updated for Android 7	Brian Wood

Table of Contents

List of Figures

1 Document Introduction

This document contains enterprise guidance for the deployment of Samsung devices in accordance with the Common Criteria configuration.

1.1 Evaluated Devices

The Common Criteria evaluation was performed on a set of devices covering a range of processors. These devices were chosen based on the commonality of their hardware across several different devices that are also claimed through equivalency. All device models are evaluated with Samsung Android 7 (Nougat).

The evaluation was performed on the following devices:

- System LSI Exynos and Qualcomm Snapdragon
  - Galaxy S7 Edge
- Qualcomm Snapdragon
  - Galaxy S8 +
  - Galaxy Tab S3
- System LSI Exynos
  - Galaxy S8
  - Galaxy S6 Edge

The following table shows the devices for which equivalence is being claimed from each evaluated device.

Evaluated Device	Processor	Equivalent Devices	Differences
Galaxy S8 + (Qualcomm)	Snapdragon 835	Galaxy S8 (Qualcomm)	S8 + is larger
		Galaxy S8 Active	S8 + is larger S8 Active has a IP68 & MIL-STD-810G certified body
Galaxy S8 (System LSI)	Exynos 8895	Galaxy S8 + (System LSI)	S8 + is larger
Galaxy Tab S3 (T825Y)	Snapdragon 820	Galaxy Tab S3	T825 & T827 models have LTE T820 models only have Wi-Fi
	Snapdragon 820	Galaxy S7 (Qualcomm)	Curved screen vs. Flat screen

Evaluated Device	Processor	Equivalent Devices	Differences
Galaxy S7 Edge (Qualcomm)		Galaxy S7 Active	Curved screen vs. Flat screen S7 Active has a IP68 & MIL-STD-810G certified body No fingerprint sensor
Galaxy S7 Edge (System LSI)	Exynos 8890	Galaxy S7 (System LSI)	Curved screen vs. Flat screen
Galaxy S6 Edge	Exynos 7420	Galaxy S6	Curved screen vs. Flat screen
		Galaxy S6 Edge+	Curved screen vs. Flat screen
		Galaxy Note 5	Curved screen vs. Flat screen Note 5 is larger Note 5 includes stylus & functionality to take advantage of it for input (not security related)
		Galaxy S6 Active	Curved screen vs. Flat screen S6 Active has a IP68 & MIL-STD-810G certified body No fingerprint sensor

The differences between the evaluated devices and the equivalent ones do not relate to security claims in the evaluated configuration. The Wi-Fi chipsets are the same for each series of common devices.

The model numbers and evaluated versions of the mobile devices being claimed are as follows:

Device Name	Base Model Number	Android Version	Kernel Version	Build Number	Carrier Models
Galaxy S8 (Qualcomm)	SM-G950	7.0	4.4.16	NRD90M	U
Galaxy S8 (System LSI)	SM-G950	7.0	4.4.13	NRD90M	N, F
Galaxy S8 + (Qualcomm)	SM-G955	7.0	4.4.16	NRD90M	U
Galaxy S8 + (System LSI)	SM-G955	7.0	4.4.13	NRD90M	N, F
Galaxy S8 Active	SM-G892	7.0	4.4.16	NRD90M	A, None
Galaxy Tab S3	SM-T820	7.0	3.18.31	NRD90M	None
	SM-T825	7.0	3.18.31	NRD90M	N, Y, None
	SM-T827	7.0	3.18.31	NRD90M	V, A, R4
Galaxy S7 (Qualcomm)	SM-G930	7.0	3.18.31	NRD90M	T, P, R4, V, A
Galaxy S7 (System LSI)	SM-G930	7.0	3.18.14	NRD90M	F, S, K, L
Galaxy S7 Edge (Qualcomm)	SM-G935	7.0	3.18.31	NRD90M	A, T, P, R4, V
Galaxy S7 Edge (System LSI)	SM-G935	7.0	3.18.14	NRD90M	F, S, K, L
Galaxy S7 Active	SM-G891	7.0	3.18.31	NRD90M	A, None
Galaxy S6 Edge+	SM-G928	7.0	3.10.61	NRD90M	F, I, A, T, P, R4, V, S, K, L
Galaxy Note 5	SM-N920	7.0	3.10.61	NRD90M	I, A, T, P, R4, V, S, K, L

Device Name	Base Model Number	Android Version	Kernel Version	Build Number	Carrier Models
Galaxy S6	SM-G920	7.0	3.10.61	NRD90M	F, I, A, T, P, R4, V, S, K, L
Galaxy S6 Edge	SM-G925	7.0	3.10.61	NRD90M	F, I, A, T, P, R4, V, S, K, L
Galaxy S6 Active	SM-G890	7.0	3.10.61	NRD90M	A, None

The Carrier Models column specifies the specific versions of the devices which have the validated configuration. These additional letters/numbers denote carrier specific models (such as V = Verizon Wireless). Only models with the suffixes listed in the table can be placed into the validated configuration.

Note: Where Carrier Models specifies "None" that means a device without a suffix is also a device which can be placed into a validated configuration.

The following table shows the Security software versions for each device.

Device Name	MDF Version	MDF Release	WLAN v1.0 Release	VPN v1.4 Release	KNOX Release
Galaxy S6, S6 Edge, S6 Active, Note 5	3.0	2	2	8.1	2.7
Galaxy S7, S7 Edge, S7 Active, Tab S3	3.0	2	2	8.1	2.7
Galaxy S8, S8+, S8 Active	3.0	2	2	8.1	2.8

The MDF version number is broken into two parts as the claimed MDFPP has been updated in the latest devices. For example, the Galaxy S8 would show "MDF v3.0 Release 2".

1.2 Terminology/Glossary

ADB	Android Debug Tool
ADT	Android Development Tools
API	Application programming interface
BYOD	Bring-Your-Own-Device
CA	Certification Authority
MDM	Mobile Device Management
ODE	On-Device Encryption
SDK	Samsung Enterprise Software Development Kit

SSL	Secure Socket Layer
VPN	Virtual Private Network

2 Guidance Overview

The Samsung model to maintain a secure mobile device environment involves a number of parties. These include:

- Approved Mobile Device Management (MDM) software developers;
- Samsung Approved Carriers;
- Enterprise and Mobile Device Administrators; and
- Enterprise Users.

As a result, a number of elements of maintaining a secure mobile environment are reliant on parties outside of Samsung and are not detailed in this documentation.

This document has been designed for Enterprise and Mobile Device Administrators and therefore provides guidance on the configuration and deployment of a Mobile Enterprise solution using Samsung devices. Guidance for device users is provided in a separate document.

3 Introduction

3.1 Overview

The TOE is a mobile operating system based on Android 7 with modifications made to increase the level of security provided to end users and enterprises. The TOE is intended to be used as part of an enterprise messaging solution providing mobile staff with enterprise connectivity.

The TOE combines with a Mobile Device Management (MDM) solution that enables the enterprise to watch, control and administer all deployed mobile devices, across multiple mobile service providers as well as facilitate secure communications through a VPN. This partnership provides a secure mobile environment that can be managed and controlled by the environment and reduce the risks that can be introduced through a Bring-Your-Own-Device (BYOD) model.

The Samsung Enterprise Software Development Kit (SDK) builds on top of the existing Android security model by expanding the current set of security configuration of options to approximately 650 configurable policies and including additional security functionality such as application blacklisting. The ability to set these policies is based on the capabilities of the MDM.

3.2 Evaluated Capabilities

The product provides a significant amount of security capabilities with the core capabilities being included within the common criteria evaluation including:

Security feature	Description
Device data protection. The TOE provides security functionality to protect data at rest.	**On Device Encryption (ODE).** The TOE has the ability to encrypt data on the device using AES 256.
	Removable storage encryption. The TOE can encrypt all file placed onto, or already reside on, removable storage attached to the device.
	Sensitive data protection. The TOE has the ability to securely store incoming data that is considered sensitive such that it can't be decrypted without the user logging in.

Security feature	Description
Application Management. The device provides a number of security functions to manage device software.	**Application resource restrictions.** All applications are run within a controlled environment that limits applications to only accessing only authorized data and resources.
Access Control. The device can implement access control that reduces mobile user permissions and assists in reducing unauthorized access.	**Device lock.** The TOE can be configured to automatically lock after a defined period of inactivity (1 to 60 minutes) limiting access to device functions accept those that are explicitly authorized such as emergency calls.
	Local wipe. The TOE has the ability to wipe encryption keys/data on a device after an administratively defined amount of authentication attempts are surpassed.
	Credential complexity. The TOE can enforce enterprise password policies forcing users to use a defined level of complexity in device passwords.
	Privileged access. The TOE can be configured to restrict mobile user's access to privileged functions such as device configurations.
	Hotspot Control. The TOE can be configured to act as a hotspot for sharing Internet access to other devices.
	Wireless network settings. The wireless network configuration of the TOE can be specified, providing requirements or pre-loaded networks.
Enterprise device management. Enterprise administrators can control and audit mobile endpoint configurations and wipe device if needed.	**Remote wipe.** An enterprise administrator can send a message to the TOE to wipe all local storage and the SD card.
	Security policy. The TOE can be configured by a Mobile Device Management solution that supports the Samsung Enterprise SDK.
	Auditing. The TOE can monitor an generate records related to security-relevant events within the device.

3.3 KNOX Management API

Samsung provides an extensive set of management APIs to fully control a Samsung device within your environment. To obtain more information about specific APIs and capabilities provided by Samsung, sign up for an account at https://seap.samsung.com/ and request access to the MDM API.

4 Deployment process

The specific deployment model is dependent on a number of factors including:

- Chosen MDM solutions supported architecture;
- Preferred mobile operating methods (often as a result of business culture);
- Financial considerations;
- Enterprise technical capability
- Risk appetite of the business; and
- Existing technological capital.

4.1 Enterprise architecture

The first step in deploying Samsung devices is to decide on both a Mobile Device Management solution and an appropriate architecture. These two selections may be done in either order depending on the preferences of the organization. In some organizations there may be a preferred architecture, and as a result an MDM solution is based on its compatibility with that architecture, in others, the architecture will be chosen to match the already chosen MDM.

There are three core architectures:

- Enterprise based deployment;
- Cloud based deployment; and
- Hybrid approach.

However, only the 'enterprise based deployment' architecture will be described in detail. The 'cloud based deployment' and the 'hybrid approach' are not covered by this evaluation, though they are certainly options which can be employed. Ideally any MDM solution will have been evaluated to the requirements of the MDMPP (Mobile Device Management Protection Profile).

4.1.1 Enterprise based deployment

In this architecture the enterprise environment must provide all of the services required to operate and manage devices. The basic components of this model include:

- **Mobile Device Management Solution**

  The Mobile Device Management (MDM) Solution secures monitors, manages and supports mobile devices deployed across companies. By controlling and protecting the data and configuration settings for all Android devices in the corporate network business security risks are reduced. Samsung offers an extensive range of different solutions. Every Mobile Device Management solution supports the Samsung Enterprise SDK.

  Android devices combine with a Mobile Device Management solution. This partnership provides a secure mobile environment that can be managed and controlled by the environment and reduce the risks that can be introduced through a Bring-Your-Own-Device (BYOD) model.

- **Secure tunnel termination**

  A secure VPN tunnel should be initialized between the managed Android devices and the Enterprise Environment to prevent unauthorized access to enterprise resources. The connection should be based on certificates deployed on the Android user devices. Ideally mutual authentication is deployed, meaning that both the Android user devices authenticate themselves with a certificate but also the gateway to the enterprise environment. Mutual authentication serves to prevent Android user devices to login into an unauthorized enterprise network and on the other hand prevents the unauthorized login of untrusted devices into the enterprise environment.

  The tunnel establishment should be terminated in case of invalid certificates. Further, an idle VPN session should be terminated after a certain time span.

- **Directory services**

  The directory services should be set up to store, organize and provide access to information in a directory.

- **Business applications**

  Business applications allow enterprise users to fulfill or access certain business tasks pertinent to requirements. This may include management tools, accounting utilities and contact management software/solutions.

- **Certificate services**

A certificate service must be implemented that manages all certificate needs throughout the enterprise environment. This includes issuing new Android device user certificates that are needed to facilitate secure communications through a VPN.

The advantages of this solution are that there will be no issue with data sovereignty plus the enterprise increases its control of the over the managed devices as well as the deployed environment. The downside is the increased costs for managing this enterprise environment.

Figure 1 shows an example of a high level design of an enterprise based environment.

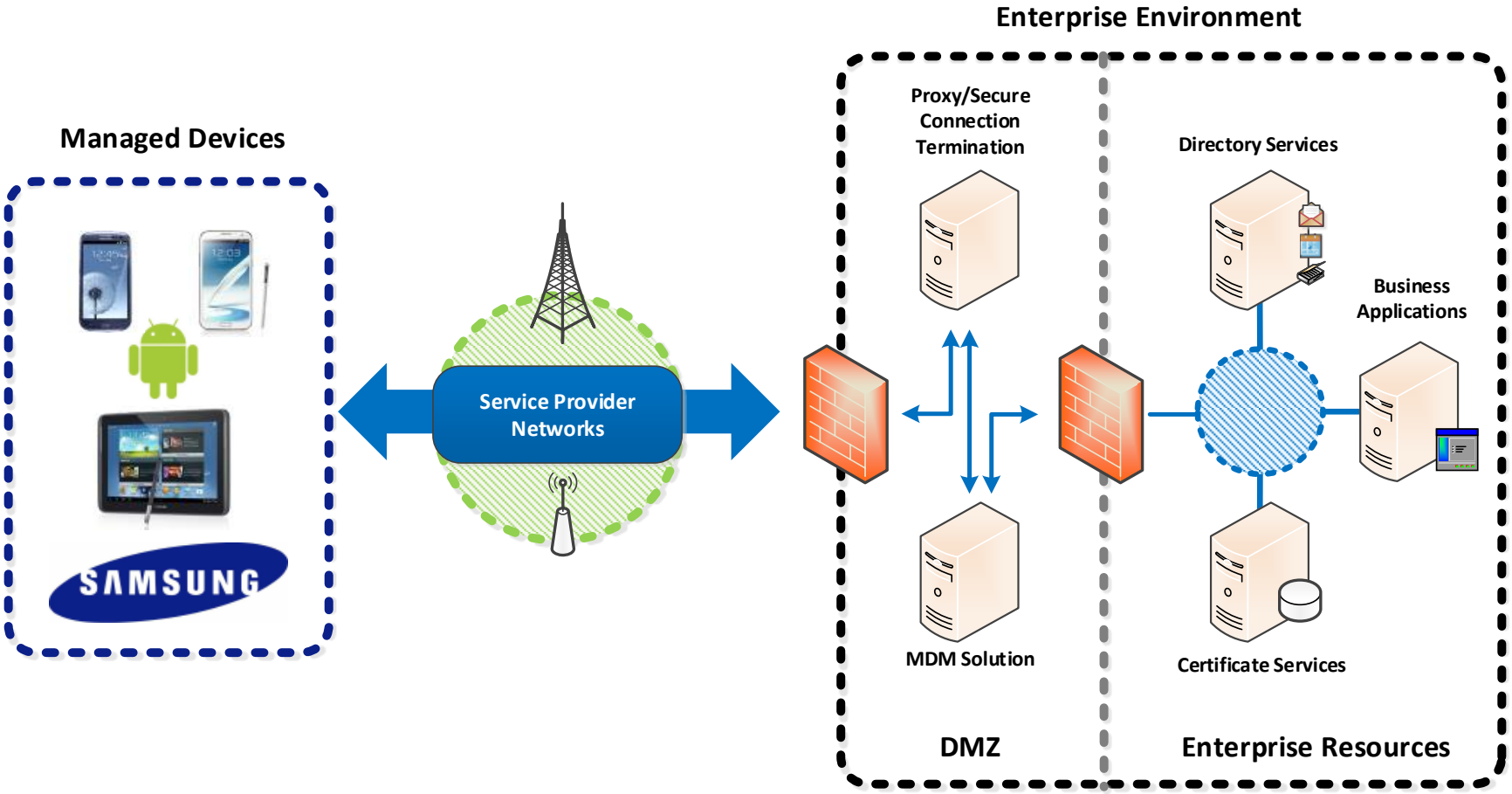

Figure 1 – Enterprise Environment

4.1.2 Compatible Mobile Device Management (MDM) solutions

The security configuration specified here can be set through the MDM for the evaluated configuration or through the installation of an application provided by Samsung and some user configuration settings as specified below. All other configuration items can be changed without changing the evaluated configuration. The evaluated configuration is provided in Section 4.3.2.

4.2 Secure preparation of the Enterprise Environment

Prior to the configuration of a Samsung Android user device, the enterprise environment must be securely prepared.

In particular, the guidance for the Mobile Device Management Solution should be followed. This documentation provides information about the capability to remotely manage devices and perform functions such as sending remote wipe messages. Further, it includes, or provides directions to implement, infrastructure to support secure transmissions with devices.

For an enterprise deployment of Samsung Android devices that is suitable for organizations working with official data, administrators should:

- Deploy and configure the requisite network components as described above
- Procure and set up an MDM server with a client that implements the KNOX APIs and is able to enforce all the settings given in the Common Criteria Configuration section below.

Section 4.3.2 provides more detailed information about the options the MDM must support in order to configure the devices in the evaluated configuration.

4.3 Secure installation of Samsung Android user devices

This section provides information on how an Enterprise and Mobile Device Administrator securely installs a Samsung Android user device.

For an enterprise deployment of Samsung Android devices that is suitable for organizations working with official data, administrators should:

- Perform the device deployment process described in on Section 4.3.1; and
- Create MDM security profiles for the devices in line with the guidance given in the Common Criteria Configuration (Section 4.3.2) and associate these profiles with the devices.

4.3.1 Device deployment process

The following steps should be followed to provision each end user device onto the enterprise network to prepare it for distribution to end users.

1. Install the MDM agent application, and enroll the device into the MDM.
2. Provision client certificates by either:
    a. Provisioning the client certificates using a locally-enrolled MDM server;
    b. Deploying the Android Development Tools (ADT) bundle and device-specific USB drivers onto a dedicated provisioning terminal. This will allow the client certificates to be manually deployed onto the device via the Android Debug Tool (ADB). Note that USB debugging should be disabled once provisioning is complete.

       The certificates required for an MDM deployment are:

        i. Enterprise CA certificate (used to validate the server certificates presented by the VPN endpoint and reverse proxy),
        ii. VPN client certificate (for authentication to the enterprise VPN endpoint),
        iii. SSL client certificate (for authentication to the reverse proxy for intranet services).
3. Install applications required for enterprise productivity.
4. Ensure that only trusted applications are installed and enabled on the device (disable unnecessary applications, including Google Play if necessary).
5. Configure on-device security settings (please refer also to Section 4.3.2).
6. Configure the VPN client to connect to the enterprise VPN endpoint, using the device-specific client certificate that has been loaded onto the device. Enable 'Always-On' VPN
7. Configure the email client to connect to the enterprise server using client certificate authentication.

4.3.2 Common Criteria Configuration

The following table shows settings which must be enabled to a specific value (or range of values) to meet the specification of the evaluation. The evaluated security configuration consists of both Samsung

specific (Samsung Enterprise SDK) as well as Android specific settings. Please also follow the guidance provided in [MDMG] to set the options listed below. The Classes or Methods used to configure these settings are provided for reference and can be used to verify whether the MDM will support your needs.

The following sections specify the required settings that must be enabled/configured to place a device into the evaluated configuration.

Note: Methods that can meet the requirement that are provided by Android natively are listed in *italics*. In most cases there is a corresponding Samsung KNOX API as well. When this is the case, the two Methods are highlighted to show the correspondence between the options. In these cases, the MDM may use either call to achieve the same result.

4.3.2.1 Configurations with and without KNOX Containers

Samsung devices include an integrated capability to create separate containers within the device. These are enabled by the KNOX components included in Samsung Android. When a KNOX container is configured it provides a separated area of the device which can have its own apps and data which is not accessible from the "normal" area. The KNOX containers can be used to separate different apps, such as in BYOD scenarios where an enterprise could place their data into a separate container on the user's device.

A Samsung device can be placed into an evaluated configuration both with and without a KNOX container. For organizations that do not need to segment the device, a configuration can be used without creating any KNOX containers. For organizations that have a need for data separation, KNOX containers can be created and still be in an evaluated configuration.

The setting listed below show the APIs which are used to place a device into an evaluated configuration for either case. When configuring a device to use a KNOX container all settings marked (All) and those marked (KNOX) are used. When using a device without a KNOX container only the settings marked (All) need to be used.

Note: KNOX containers implement many of the same APIs as are available to non-containers (such as hardware state configurations). Policies in KNOX containers are tied specifically to those containers as part of the KNOX Premium API configuration. All KNOX APIs specified are part of the KNOX Premium set of APIs and require a KNOX license to be used.

4.3.2.2 CC Mode Settings (All)

To place a device into the evaluated configuration the CC Mode must be enabled.

Setting	Value	Description	Class or Method
CC Mode	Enable/ Disable	This setting enables FIPS-validated crypto, disables USB connectivity in recovery mode & only allows FOTA updates to the system	setCCMode()

To ensure overall control of the configuration, once enabled, CC Mode cannot be disabled by an end user except by performing a factory reset. It is possible to change the CC Mode status through the MDM, a user can only turn off CC Mode by choosing to perform a factory reset.

4.3.2.2.1 CC Mode without MDM Support

CC Mode is not yet widely supported by MDM vendors. To facilitate customers in enabling CC Mode, Samsung has provided a stand-alone app that can enable this setting locally on the device.

The CCMode.apk can be downloaded from Samsung here. In addition to the APK, you can download the latest guidance documentation and the list of applications provided with each validated device.

For full instructions on configuring the device into the CC Mode utilizing the application, review the Common Criteria User Guidance Documentation for the device that can be found at the same website.

4.3.2.2.2 CC Mode and Approved Cryptography

Part of the Common Criteria-evaluated configuration is the availability of approved cryptographic engines for use by the system and applications. Samsung has chosen to utilize FIPS 140-2-validated cryptographic modules on its devices for the Common Criteria configuration.

Samsung provides the following cryptographic modules on all the evaluated devices:

- Samsung Kernel Cryptographic Module
- BoringSSL FIPS Object Module
- SCrypto Module

All modules always run in a FIPS-validated mode. BoringSSL, for compatibility reasons, provides access to non-FIPS algorithms, which developers should not utilize in a validated configuration (but which are necessary to ensure functionality with many commercial services). The APIs which provide access to FIPS-validated algorithms are detailed in the User Guidance documentation.

Note: Only these modules have been evaluated. It is also possible that some applications may implement their own cryptography. Only the cryptographic modules provided with the device are validated, any other cryptography must be evaluated on its own. Samsung recommends that developers utilize the cryptographic functions provided with the device.

4.3.2.2.3 CC Mode Status

CC Mode has the following statuses:

Status	Description
Ready (blank)	CC Mode has not been turned on
Enforced	CC Mode has been turned on but some of the required settings or configurations have not been set
Enabled	CC Mode has been turned on and all required settings and configurations have been set
Disabled	CC Mode has been turned on but an integrity check or self-test has failed (such as a FIPS 140-2 self-test)

The CC Mode status can be seen by going to ***Settings/About phone/Software Security Version***. Clicking on the item will show the current status.

Note: The Ready state does not have any indicator. Only Enforced, Enabled and Disabled actually show a specific status

4.3.2.2.4 CC Mode Requirements/Configurations

When CC Mode is first turned on, it changes the status from Ready to Enforced. To change the status to Enabled, the following settings must be configured:

1. Enable the Maximum Password Failure Policy
2. Enable On Device Encryption (ODE)
3. Enable SD Card Encryption
4. Enable CRL Checking

Note: To be Enabled, not only must the encryption settings be set, the storage must have been encrypted by the user.

4.3.2.3 Encryption Settings (All)

There are two sets of encryption settings, one for internal storage and one for external (SD Card) storage. Both must be enabled, even if no SD Card will be used in the device.

Setting	Value	Description	Class or Method
On Device Encryption (ODE)	Enable	This encrypts all internal storage media	setInternalStorageEncryption() *setStorageEncryption()* setRequireDeviceEncryption()
SD Card Encryption	Enable	This encrypts all external (SD Card) storage media	setExternalStorageEncryption() setRequireStorageCardEncryption()

4.3.2.4 Authentication Settings (All)

Setting	Value	Description	Class or Method
Max Password Failures (Wipe)	10 or less for device 99 or less for KNOX	The maximum number of times a password can be entered before the device is wiped	*setMaximumFailedPasswordsForWipe()*

4.3.2.5 Certificate Revocation Settings (All)

Setting	Value	Description	Class or Method
Certificate Revocation Checking	Enable for All apps	Specifies that CRL checking is enabled for all apps on the device	enableRevocationCheck()

4.3.2.6 KNOX Container Policy (KNOX)

Setting	Value	Description	Class or Method
Create Container	Policy of container	Specifies the policy to be used when creating the container	Class: KnoxContainerManager createContainer() Class: CreationParams setPasswordResetToken() (a value MUST be specified here)
Configure Container Policy	Container Type and policy settings 1-99 for failed password settings	Creates a policy template for a container. This is the default settings for a new container	Class: KnoxConfigurationType KnoxConfigurationType() setMaximumTimeToLock() setPasswordMinimumLength() setPasswordQuality() setPasswordMinimumSymbols() setMaximumFailedPasswordsForWipe() setMaximumFailedPasswordsForDeviceDisable() addConfigurationType()
Remove/Wipe Container	Container ID	Removes the specified container and wipes all data associated with that container	removeContainer()

Note: When set in a KnoxConfigurationType(), the setMaximumFailedPasswordsForDeviceDisable() or the setMaximumFailedPasswordsForWipe() settings will disable or wipe the container, not the whole device.

4.3.3 Other Common Criteria Configurations

The settings in this section have been evaluated, but no specific configuration is required to place the device into the evaluated configuration. They are part of the management functions that are included and can be configured as needed for your specific environment.

4.3.3.1 Authentication Settings (All)

The settings here deal with passwords and other authentication-related settings.

Setting	Value	Description	Class or Method
Password Length	6-16	Minimum password length	*setPasswordMinimumLength()*
Password Complexity	Set min # of characters or max sequences	Settings to require different types of characters in a password	*setPasswordQuality()* setMaximumCharacterOccurrences() setMaximumCharacterSequenceLength() setMaximumNumericSequenceLength() setMinPasswordComplexChars() setMinimumCharacterChangeLength() *setPasswordMinimumLetters()* *setPasswordMinimumLowerCase()* *setPasswordMinimumNonLetter()* *setPasswordMinimumNumeric()* *setPasswordMinimumSymbols()* *setPasswordMinimumUpperCase()*
Password Expiration		Specify the maximum age of a password before it must be changed	setPasswordExpires() *setPasswordExpirationTimeout()*
Password Entry Visible	Disable	This prevents entered passwords from being displayed on the screen	setPasswordVisibilityEnabled() setScreenLockPatternVisibilityEnabled()

Setting	Value	Description	Class or Method
Biometrics Use	Enable/ Disable	Enable or Disable biometric authentication methods (fingerprint or iris, depending on device support)	setBiometricAuthenticationEnabled()

Note: To control biometrics usage with setBiometricAuthenticationEnabled(), setPasswordQuality() must be set to something other than PASSWORD_QUALITY_UNSPECIFIED (which is the default setting).

Biometrics are not supported as a stand-alone authentication method for the KNOX Container. Biometrics can only be used as part of a hybrid or multifactor authentication method.

4.3.3.2 Admin Settings (All)

Setting	Value	Description	Class or Method
Allow new admin	Enable / Disable	An MDM can prevent new admins from getting installed / activated on the device	Class: AdvancedRestrictionPolicy preventNewAdminActivation() preventNewAdminInstallation()
Prevent Admin Removal	Enable / Disable	Can be used to prevent the removal of Admin/Device Manager (i.e. MDM Agent or similar)	setAdminRemovable()
Multi-User Mode	Enable/ Disable	Allows device to support multiple, separate users (Tablets-only)	allowMultipleUsers()

Note: The preventNewAdminActivation setting requires that only one Admin/Device Manager be active when enabled. This will prevent further Admin activations. If two (or more) are already enabled, the setting will be ignored.

4.3.3.3 Lock screen Settings (All)

Setting	Value	Description	Class or Method
Inactivity Timeout Lock Period	1 to 60 minutes	This specifies how long the device will remain unlocked after usage has stopped	setPasswordLockDelay() *setMaximumTimeToLock()*
Remote Lock	Enable	Will remotely lock the device immediately	*lockNow()*
Unlock Banner	Up to 256 characters	Text to display on the lock screen before login. Text will scroll at XX characters	changeLockScreenString()

4.3.3.4 Lock screen Settings (Device only)

Setting	Value	Description	Class or Method
Lock screen controls	Enable/ Disable lock screen features	Provides control over lock screen access to widgets/apps/ features	*setKeyguardDisabledFeatures()* *KEYGUARD_DISABLE_WIDGETS_ALL* *KEYGUARD_DISABLE_SECURE_CAMERA* *KEYGUARD_DISABLE_FEATURES_ALL*
Smart Lock Controls	Enable/ Disable	Provides control over Smart Lock capabilities (which allow authentication to be bypassed when certain conditions are met)	*setKeyguardDisabledFeatures()* *KEYGUARD_DISABLE_TRUST_AGENTS*

4.3.3.5 Radio Control Settings (All)

Setting	Value	Description	Class or Method
Control Bluetooth	Enable/ Disable	Enable or Disable access to Bluetooth	allowBluetooth()

Setting	Value	Description	Class or Method
Control Beam	Enable/ Disable	Enable or Disable Android Beam	allowAndroidBeam()
Control Wi-Fi	Enable/ Disable	Enable or Disable access to Wi-Fi	allowWiFi()
Control NFC	Enable/ Disable	Enable or Disable access to NFC	setEnableNFC()
Control Cellular Data Access	Enable/ Disable	Enable or Disable access to Cellular Data (not Voice)	setCellularData()
Control Location Provider	Enable/ Disable	Enable or Disable access to Location services on the device	setLocationProviderState()

4.3.3.6 Bluetooth Settings (All)

Setting	Value	Description	Class or Method
Bluetooth Name	Name	Set the friendly Bluetooth name of the device	*bluetoothAdapter.setName()*
Allowed Profiles	Enable/ Disable Profile	Enable or Disable the specified Bluetooth profile	setProfileState()

4.3.3.7 Wi-Fi Settings (All)

Setting	Value	Description	Class or Method
Specify SSIDs for Wi-Fi	SSID values	Enable/Disable Wi-Fi restrictions based on acceptable SSID values. Both white and black listing of networks is supported.	activateWifiSsidRestriction() addBlockedNetwork() addWifiSsidsToBlackList() addWifiSsidsToWhiteList()
Set WLAN CA Certificate	CA Cert	Specify trusted CAs for accepting WLAN server certificates	setNetworkCaCertificate()
Set Wi-Fi security type	WLAN security	Specify the type of security required on a WLAN connection (i.e. open, WEP, WPA, etc)	setMinimumRequiredSecurity()
Wi-Fi authentication protocols	WLAN security	Specify the values required to connect to EAP-TLS connections	setNetworkAnonymousIdValue() setNetworkClientCertificate() setNetworkIdentityValue() setNetworkPhase2() setTlsCertificateSecurityLevel()
Wi-Fi client credentials	WLAN client credentials	Specify the client credentials to access a specified WLAN	setNetworkPSK() setNetworkPassword() setNetworkClientCertificate() setNetworkPrivateKey() setNetworkWEPKey1-4() setNetworkWEPKeyId()

Note: In CC Mode, LEAP, PEAP and FAST modes are disabled due to their use of non-FIPS algorithms.

4.3.3.8 Hotspot/Tethering Settings (All)

Setting	Value	Description	Class or Method
Specify if Wi-Fi Hotspot can be modified by user	Enable/ Disable	Enable/Disable whether the user can edit the Hotspot settings	isWifiApSettingUserModificationAllowed()
Specify Hotspot settings	SSID, Security Type, Password	Specify the settings for the Hotspot	setWifiApSetting()
Tethering (Wi-Fi, USB and Bluetooth)	Enable/ Disable	Controls ability to use the device as a Wi-Fi hotspot to share its Internet connection. setTethering() controls access to all other tethering options (if that is disabled no others are allowed).	setTethering() setBluetoothTethering() setUsbTethering()

4.3.3.9 Services Control Settings (All)

Setting	Value	Description	Class or Method
Camera control	Enable/ Disable	Enable or Disable access to Camera	setCameraState() *setCameraDisabled()*
Microphone control	Enable/ Disable	Enable or Disable access to microphone	setMicrophoneState()

Setting	Value	Description	Class or Method
Voice control	Enable/ Disable	Enable or Disable access to S-Voice or the Voice Dialer controls. This does not prevent access to other voice-controlled apps, only the Samsung-provided ones.	allowSVoice() disableVoiceDialer() allowVoiceDialer()
Allow FOTA Updates	Enable/ Disable	Specifies whether the device can check and receive OTA updates. This can be used to block auto-updates until they have been approved.	allowOTAUpgrade()
Allow Mounting over USB Allow Locally-connected Backup	Enable/ Disable	Enable or Disable the mounting of device storage over USB. When disabled USB can only be used for charging. Locally-connected backups can only be performed when USB storage mounting is enabled.	setUsbMediaPlayerAvailability()
Allow SD Card to be mounted	Enable/ Disable	Enable or Disable the ability to mount (and use) an SD Card. Also possible to mount SD Card as read-only.	setSdCardState() allowSDCardWrite()

Setting	Value	Description	Class or Method
Allow USB storage to be mounted	Enable/ Disable	Enable or Disable the ability to mount (and use) storage through the USB port on the device (i.e. allowing a USB thumb drive to be mounted as storage to the device).	allowUsbHostStorage()
Developer Mode	Enable/ Disable	Enable or Disable USB Debugging for developer access	allowDeveloperMode() setUsbDebuggingEnabled()
Setting Automatic Time	Enable/ Disable	Enable or Disable the use of Carrier Time on the device. If this is disabled, then the time is handled solely on the device with no external checks.	setAutomaticTime()
Allow User Time Change	Enable/ Disable	Enable or Disable the ability for the user to access and change the date/time settings on the device.	setDateTimeChangeEnabled()
Google Backup	Enable/ Disable	Enable or Disable Google backup of account and settings information	setBackup()

Setting	Value	Description	Class or Method
Google Account Sync	Enable/ Disable	Enable or Disable Google account sync settings (all Google sync)	allowGoogleAccountsAutoSync()

4.3.3.10 Notification Settings (All)

Setting	Value	Description	Class or Method
Blacklist Application Notification Mode	Block All, Block Text, Block Text & Sound	Specify if notifications are blocked, what level of blocking should be done. Block all notifications, Text (status bar) notifications only, or only Text & Sound.	setApplicationNotificationMode()
App Notification Lists	App Names	Whitelist and Blacklist of apps that can override default notifications. Blacklist apps follow that notification mode setting	addPackagesToNotificationBlackList() removePackagesFromNotificationBlackList() addPackagesToNotificationWhiteList() removePackagesToNotificationWhiteList()

4.3.3.11 Lock screen Notification Settings (Device only)

Setting	Value	Description	Class or Method
Lock screen controls	Enable/ Disable lock screen features	Provides control over lock screen notifications	*setKeyguardDisabledFeatures()* *KEYGUARD_DISABLE_SECURE_NOTIFICATIONS* *KEYGUARD_DISABLE_UNREDACTED_NOTIFICATIONS*

4.3.3.12 Messaging (SMS) Settings (All)

Setting	Value	Description	Class or Method
Allow incoming messages	Enable/ Disable	Allow user to receive incoming SMS/MMS messages	allowIncomingMms() allowIncomingSms()
Allow outgoing messages	Enable/ Disable	Allow user to send SMS/MMS messages	allowOutgoingMms() allowOutgoingSms()

4.3.3.13 Certificate/Key Management Settings (All)

Setting	Value	Description	Class or Method
Import Certificates	Certs	Import CA Certificates into the Trust Anchor Database or the credential storage. The choice of storage is dependent on the type of certificate being imported.	installCertificate() installCertificatesFromSdCard() installCertificateWithType() installClientCertificate() (for VPN)
Remove Individual Certificates	Cert names	Remove Individual certificates from the database or credential store	removeCertificate()
Remove All Certificates		This will clear all imported Certificates (except the built-in TAD)	clearInstalledCertificates()

4.3.3.14 Application Management Settings (All)

Setting	Value	Description	Class or Method
Install Apps	App name	This allows an application to be installed on the device	installApplication()
Uninstall Apps	App name	This allows applications to be uninstalled from the device.	uninstallApplication() uninstallApplications() (bulk list of apps at one time)
Control app uninstall	App name	Enables / disables user uninstall of specified application	Class: ApplicationPolicy setApplicationUninstallationDisabled() setApplicationUninstallationEnabled()
Control Google Play	Enable / Disable	Allows installation of applications from Google Play	public class ApplicationPolicy enableAndroidMarket() disableAndroidMarket()
Control Unknown Sources	Enable / Disable	Allows installation of application from unknown sources	Class: RestrictionPolicy setAllowNonMarketApps
Disable Apps	Enable/ Disable	This allows an application to be disabled, even if it is installed, and prevent it from running (includes pre-installed apps)	setDisableApplication()

Setting	Value	Description	Class or Method
Application Whitelist	App name	Allows the creation of a list of approved apps that can be installed. This should always be paired with a black list.	addAppPackageNameToWhiteList()
Application Blacklist	App name	Allows the creation of a list of apps which cannot be installed.	addAppPackageNameToBlackList()
Application Signature Whitelist	App signature	Allows the creation of a list of approved signatures for apps that can be installed. This should be paired with the signature black list.	addAppSignatureToWhiteList()
Application Signature Blacklist	App signature	Allows the creation of a list of apps based on signatures which cannot be installed.	addAppSignatureToBlackList()

White/Black listing is done using the full name of the application (such as com.android.testingapp).

The method for configuring these lists is highly dependent on the MDM solution chosen. Please refer to the MDM specific guidance [MDMG] on exactly how to set these policies.

Note: The Application White/Black lists will not have any impact on apps that are part of the system image. Built-in apps can instead be Disabled.

4.3.3.15 Remote Wipe Settings (All)

Setting	Value	Description	Class or Method
Remotely wipe the device	True	Remotely wipe the data stored on the device. This will perform a factory reset.	wipeDevice() *wipeData()*

4.3.3.16 Lock Screen Settings (KNOX)

Setting	Value	Description	Class or Method
Password Length	6-16	Minimum container password length	setPasswordMinimumLength()
Password Complexity	Set min # of characters or max sequences	Settings to require different types of characters in the container password	setMaximumCharacterOccurrences() setMaximumCharacterSequenceLength() setMaximumNumericSequenceLength() setMinPasswordComplexChars() setMinimumCharacterChangeLength()
Password Expiration	(in seconds)	Specify the maximum age of the container password before it must be changed	setPasswordExpires()
Password Entry Visible	Disable	This prevents entered passwords from being displayed on the container lock screen	setPasswordVisibilityEnabled() setScreenLockPatternVisibilityEnabled()

4.3.3.17 Container Authentication Settings (KNOX)

Setting	Value	Description	Class or Method
Inactivity Timeout Lock Period	1 to 60 minutes	This specifies how long the container will remain unlocked after container usage has stopped	setPasswordLockDelay()
Remote Lock	Enable	Will remotely lock the container immediately	lock()
Multi-Factor Authentication	Enable/ Disable	Will require the user to provide both biometrics and PIN/password to unlock the container	enforceMultifactorAuthentication()

4.3.3.18 Services Control Settings (KNOX)

Setting	Value	Description	Class or Method
Camera control	Enable/ Disable	Enable or Disable access to Camera inside the container	setCameraState()
Microphone control	Enable/ Disable	Enable or Disable access to microphone inside the container	setMicrophoneState()

4.3.3.19 Application Management Settings (KNOX)

Setting	Value	Description	Class or Method
Install Apps	App name	This allows an application to be installed to the container	installApplication()

Setting	Value	Description	Class or Method
Uninstall Apps	App name	This allows applications to be uninstalled from the container.	uninstallApplication() uninstallApplications() (bulk list of apps at one time)
Control app uninstall	App name	Enables / disables user uninstall of specified application from the container	Class: ApplicationPolicy setApplicationUninstallationDisabled() setApplicationUninstallationEnabled()
Disable Apps	Enable/ Disable	This allows an application to be disabled, even if it is installed, and prevent it from running (includes pre-installed apps)	setDisableApplication()

4.3.3.20 Notification Settings (KNOX)

Setting	Value	Description	Class or Method
Blacklist Application Notification Mode	Block All, Block Text, Block Text & Sound	Specify if container notifications are blocked, what level of blocking should be done. Block all notifications, Text (status bar) notifications only, or only Text & Sound.	setApplicationNotificationMode()

Setting	Value	Description	Class or Method
App Notification Lists	App Names	Whitelist and Blacklist of apps that can override default container notifications. Blacklist apps follow that notification mode setting	addPackagesToNotificationBlackList() removePackagesFromNotificationBlackList() addPackagesToNotificationWhiteList() removePackagesToNotificationWhiteList()
Email notifications	Enable/ Disable	Sets whether the email app notifications are displayed for the container	setEmailNotificationsState()

4.3.3.21 Container Sharing Settings (KNOX)

Setting	Value	Description	Class or Method
Application & File movement	Enable/ Disable	Define whether apps can be moved into or out of the Container	allowMoveAppsToContainer() allowMoveFilesToContainer() allowMoveFilesToOwner()
Application Data Sync	Enable/ Disable for apps	Define whether specific apps can share data between the container & outside	setAllowChangeDataSyncPolicy() • Contacts • Calendar • Notifications
Clipboard sharing from Container to OS	Enable/ Disable	When enabled, this allows clipboard data to be shared outside the container	allowShareClipboardDataToOwner()

4.3.3.22 Application Control (All)

Setting	Value	Description	Class or Method
Browser Auto Text Completion	Enable/ Disable	Specifies whether the browser should automatically fill forms from stored data	setAutoFillSetting()
Control Allowed Email Accounts	Enable/ Disable for apps	Specifies restrictions on allowed email accounts. Can specify allowed or blocked domains or individual accounts.	addAccountsToAdditionBlackList() addAccountsToAdditionWhiteList() (specify restrictions for all account types)

4.3.3.23 VPN Configuration (Device Only)

The built-in Samsung VPN client can be configured for use by the whole device. More information about the specific management APIs can be found in the Samsung VPN Client on Galaxy Devices Guidance Documentation v3.0.

4.3.3.24 KNOX VPN Services (All)

KNOX provides a highly flexible method for configuring VPNs that can include the ability to control access to applications or groups of applications to specific tunnels. The KNOX VPN service can be used to control tunnels both inside and outside the container, depending on where the VPN client is installed (inside or outside the container)

To use the KNOX VPN services, the following is needed:

Component	Contents	Description
VPN Installer(s)	APKs from vendor	Installation package(s) from the VPN client vendor for installation on the device. Generally (though not always) this would include 2 files.
VPN profile(s)	json files	The VPN profile(s) to be deployed on the device

Component	Contents	Description
“vpn” folder	json files and vendor.ini	The full set of configurations (including Knox configuration) needed for deployment of the VPN profile

The VPN client vendor would provide the files above though the json configuration would have to be edited by the Administrator. More information about the json configuration can be found here: https://seap.samsung.com/api-references/android-premium/reference/com/sec/enterprise/knox/profile_creation.pdf.

A KNOX containerized VPN is a VPN installed specifically inside the KNOX container. It is possible to set multiple VPN configurations in parallel for different apps or groups of apps. It is also possible to configure dual layer VPN tunnels by using one VPN outside the container paired with a KNOX containerized VPN. These Methods are all part of the GenericVpnPolicy Class.

Setting	Value	Description	Class or Method
Create VPN Profile	VPN Vendor, json file	Specifies the VPN client vendor and the json configuration file	createVpnProfile()
Active VPN Profile	Enable/ Disable Profile	Specifies to activate or deactivate VPN profile	activateVpnProfile()
Remove VPN Profile	VPN profile	Deletes VPN profile	removeVpnProfile()
Add Apps to VPN	Package names	Adds apps to VPN profile such that these apps must use this VPN profile for connectivity	addPackagesToVpn() addAllPackagesToVpn()

Setting	Value	Description	Class or Method
Add Apps to Container VPN	Package names	Adds apps to Container VPN profile such that these apps must use this VPN profile for connectivity	addContainerPackagesToVpn() addAllContainerPackagesToVpn()
Remove Apps to VPN	Package names	Removes apps from VPN profile	removePackagesFromVpn() removeAllPackagesFromVpn()
Remove Apps to Container VPN	Package names	Removes apps from Container VPN profile	removeContainerPackagesFromVpn() removeAllContainerPackagesFromVpn()

Note: When adding packages to a VPN profile, use User0 for the whole device and User100 for the Knox container.

4.3.4 Sensitive Data Protection

Samsung has added capabilities for Sensitive Data Protection. This feature is designed to allow applications which run in the background and receive information to protect that information upon receipt. This feature is provided as part of the device, but its use is dependent on applications having been written to the APIs providing the capability. It is expected that this list will grow over time, but is currently limited to the Samsung Email application contained within the KNOX container.

The API for Sensitive Data Protection exists both for the whole device and KNOX, but unless an application has been written to the API, it will not take advantage of the Sensitive Data Protection function.

4.3.5 Additional notes

Samsung Android devices are usually configured by default to send anonymous usage data (including location, device ID etc.) to Google and Samsung servers. This can be disabled through device settings and will need to be enforced through procedural controls.

Samsung Android devices do not need to be associated with a Google account to operate as required within the enterprise. For example, it is still possible to receive push notifications through Google Cloud

Messaging. KNOX MDM APIs can be used to prevent users from signing in to these services (see [MDMG]).

4.4 Audit Records (KNOX)

Auditing is enabled and events retrieved through the MDM. A KNOX Premium license is required in order to enable the collection of audit records.

Audit records are stored in a compressed format to minimize space and maximize the amount of records that can be stored. When the allocated space is full the oldest events will be overwritten so the most recent as always maintained (circular logging/buffering). Notifications are sent to the MDM based on the log space becoming full to warn before wrapping occurs.

The minimum amount of allocated space for audit storage is 10MB with a maximum of 50MB, depending on the available free space when activated. There must be at least 200MB of free space when Auditing is enabled (an error is returned to the MDM if not), and no more than 5% of free space will be used, up to the maximum of 50MB. The allocated space is not adjusted after it is initially set.

Within the logging it is also possible to specifically filter the events that are written to the log.

4.4.1 Types of Audit Events

There are three classes of audit events that can be logged, system and apps, kernel and IP tables. Each can be controlled individually, so you can log just select classes of events. Kernel and IP table logging generates a large amount of events, so care should be taken that the MDM collect the logs frequently if they are enabled or the circular logging function could cause events to be overwritten and lost.

4.4.2 Audit Collection Settings

All methods are in the class com.sec.enterprise.knox.auditlog.

Setting	Value	Description	Class or Method
Enable Auditing	-	Enables audit collection	enableAuditLog()
Disable Auditing	-	Disables audit collection	disableAuditLog()

Setting	Value	Description	Class or Method
Configure Logging Filters	See Filter Settings table	Configures what events to be captured (see Filter table)	setAuditLogRules()
Enable IP Tables Auditing	-	Enables the collection of IP Tables	enableIPTablesLogging()
Disable IP Tables Auditing	-	Disables the collection of IP Tables	disableIPTablesLogging()

4.4.2.1 Audit Collection Filter Settings

Member	Values	Values & Description
setSeverityRule(int severityRule)	1 = Alert 2 = Critical 3 = Error 4 = Warning 5 = Notice	Specifies the minimum severity level to log. Everything with the specified number and lower will be logged.
setOutcomeRule(int outcomeRule)	0 = Fail 1 = Success 2 = All	Specifies filtering based on the outcomes of each event
setGroupsRule(List<Integer> groupsRule)	1 = Security 2 = System 3 = Network 4 = Events 5 = Application NULL = All	Specifies the groups of events to log. NULL will log events from all groups.
setKernelLogsEnabled(boolean enableKernel)	True = Enable False = Disable	Enables or disables Kernel logging
setUsersRule(List<Integer> usersRule)	List of UID	This allows logging only from specified UIDs in the list. This is only available to MDMs outside the KNOX container (inside the container the MDM can only see the container user). System events (group 2) are always logged.

4.4.3 Audit Record Fields

The audit records have eight (8) fields as described in the following table.

Field	Description
Timestamp	Long value that represents the UTC timestamp
Severity	Integer value representing the severity: 1 (alert), 2 (critical), 3 (error), 4 (warning), 5 (notice)
Group	Integer value representing the group code: 1 (security), 2 (system), 3 (network), 4 (events), 5 (application)
Outcome	Integer value representing the outcome of the event: 1 (success), 0 (failure)
PID	Integer value representing the process ID
USERID	Integer value representing the USERID for which the log was originated ID 0 is for a normal user ID -1 is for system events ID 100-102 is for container users (multiple containers can be defined)
Component	String representing the facility/Software Component name
Message	Free-form message description of the event (generally a human-readable message)

4.4.4 Audit Events and Management

One important note about the audit capabilities is that they are tied to being enrolled to a management server (MDM/EDM). If the device is not enrolled there is no way to enable auditing, and when a device is unenrolled, the audit records are deleted as part of the unenrollment process, so any events created between the last review/upload and the unenrollment will be lost.

4.4.5 Audit Events

The following list of audit records are produced related to the functionality claimed in the MDFPP.

4.4.5.1 System and Audit Service Events

Message	Description
AuditLog status has changed to <*enable*>	This shows the status of the audit log. Note that when disabled, audit logs are erased
AuditLog has reached its critical size. Percentage is <*value*>	Shows the audit storage has reached the percentage full set in <*value*>
Android boot completed	Startup of the operating system is complete
Android will be shutdown	A shutdown command has been sent to the device (from any source)
AuditLog filter rules has changed	The filtering rules for the audit log have been changed

4.4.5.2 Common Criteria Mode Events

Message	Description
Admin <*admin pkg name*> has requested to <*enable, disable*> CCMode	Shows when CC Mode is enabled or disabled by policy
CC Mode status : <*Ready, Enforcing, Enabled, Disabled*>	Shows the current CC Mode status on device startup

4.4.5.3 Encryption-related Events

These events cover ODE, external media encryption and FDP_DAR_EXT.2 functions.

Message	Description
Admin <*admin pkg name*> has requested storage encryption	These messages show the MDM enabling ODE. The setting is requested, then the process of encryption is requested, and lastly showing it is active after the encryption process is completed.
Admin <*admin pkg name*> has requested SD card encryption.	These messages show the MDM enabling SD card encryption.
Admin <*admin pkg name*> has requested encryption of external storage	These messages show the MDM enabling external storage encryption (such as when media is plugged in through the USB port).
Encrypting storage card <*succeeded/failed*>	Shows success and failure of storage encryption for the SD Card.
user_id[<*uid*>]/pid[<*pid*>] failed to access file [<*file*>]	Shows failure of storage services related to FDP_DAR_EXT.2

4.4.5.4 Administration Settings

Message	Description
Admin <*pkg name*> has changed maximum failed passwords for wipe to <*value*>	Shows the maximum failed passwords allowed before the device is wiped has been set to <*value*>
Admin <*admin pkg name*> has changed password minimum length to <*value*>	The minimum password length has been set to <*value*>
Admin <*admin pkg name*> has changed password quality to <*value*>	The password quality (complexity) has been set to <*value*>
Admin <*admin_pkg*> has changed screen lock time out to <*value msec*>	The session timeout for locking the screen has been set to <*value msec*>
Admin <*admin_pkg*> has changed maximum failed passwords for disable to <*value*>	The maximum failed password attempts before locking the device is set to <*value*>

Message	Description
Admin <*admin_pkg*> has changed maximum character occurrences to <*value*>	The maximum number of times the same character can be in the password is set to <*value*>
Admin <*admin_pkg*> has changed password expiration time out to <*value msec*>	The password expiration has been set to <*value msec*>
Admin <*uid*> has <*allowed|disallowed*> <*biometric type*>.	The admin has allowed or blocked access to the specific biometric listed: • BIOMETRIC_AUTHENTICATION_FINGERPRINT • BIOMETRIC_AUTHENTICATION_IRIS
Admin <*admin_pkg*> has <*enabled, disabled*> reboot banner [with text <*text*>] Lock screen string was enabled. Lock screen string was changed to <*value*> Admin <*admin*> has cleared the lock screen string	Setting of the login banner

Message	Description
Admin <*admin_pkg*> has <*allowed|enabled, disallowed|disabled*> <*feature*>	The admin has allowed or disallowed the following features: • Camera • Microphone • Developer mode • Airplane mode • USB Tethering setting • Wi-Fi Tethering setting • Bluetooth tethering • NFC • Cellular data • USB debugging • USB Media Player (MTP) • VPN • S-Beam • Android Beam • S-Voice • USB Host Storage • <profile> Bluetooth profile
Admin <*admin_pkg*> has changed WiFi allowed to <*true, false*>	Admin has enabled or disabled Wi-Fi
Admin <*admin_pkg*> has changed allow bluetooth to <*true, false*>	Admin has enabled or disabled Bluetooth

4.4.5.5 KNOX Container Management

Most of the management functions for the container (such as password management or camera access) generate the same messages as outside the container. The messages inside the container will be marked with the Container ID (usually 100).

Message	Description
Admin <*admin pkg name*> has successfully requested to create container.	Creation of a KNOX Container has been requested
Admin <*admin pkg name*> has successfully removed container <*container ID*>	Removal of a KNOX Container.
Admin <*admin pkg name*> has <*locked/unlocked*> container	The admin has locked or unlocked the container

4.4.5.6 KNOX Container Sharing

Message	Description
Admin <*uid*> has <*allowed|disallowed*> moving applications to container.	The admin has allowed or disallowed applications to be moved to the container.
Admin <*uid*> has <*allowed|disallowed*> moving files to container.	The admin has allowed or disallowed files to be moved to the container.
Admin <*uid*> has <*allowed|disallowed*> moving files to owner.	The admin has allowed or disallowed files to be moved from the container.
Admin <*uid*> has <*allowed|disallowed*> sharing clipboard to owner from container.	The admin has allowed or disallowed the clipboard to be shared from the container.

4.4.5.7 External Media Access

Message	Description
Admin <*admin pkg name*> has <*enabled/disabled*> access to external SDCard	Shows when SD card mounting has been allowed or blocked

Message	Description
Removable Media Event : External SD Card Mounted Removable Media Event : External SD Card Unmounted	Shows mount events for SD card and USB media.

4.4.5.8 Factory Reset Events

These events are only for reset failures since by definition a successful reset would wipe out the audit logs.

Message	Description
Starting user data Wipe	This event is seen when a wipe fails to occur and the logs are not wiped.

4.4.5.9 Administration Command Events

Message	Description
Admin <*admin pkg name*> has locked device	The admin has forced the device to lock
Admin <*admin pkg name*> has requested full wipe of device	The admin has sent a command to force a factory reset

4.4.5.10 Key Management Events

Message	Description
Key generation failed, with error <*number*>	The generation of a key pair failed

Message	Description
Key importing activity (Keystore=<keystore>, keyName=<*keyname*>, uid=<*target uid*>, requested by <*pkg name*>: uid=<*uid*> role=<*SystemApp, UserApp*> | <*Administrator, NonAdministrator*>) succeeded Key importing activity (Keystore=<keystore>, keyName=<*keyname*>, uid=<*target uid*>, requested by <*pkg name*>: uid=<*uid*> role=<*SystemApp, UserApp*> | <*Administrator, NonAdministrator*>) failed with error <*error msg*>	Shows success or failure of importing keys to the keystore.
Key destruction activity (Keystore=<keystore>, keyName=<*keyname*>, uid=<*target uid*>, requested by <*pkg name*>: uid=<*uid*> role=<*SystemApp, UserApp*> | <*Administrator, NonAdministrator*>) succeeded Key destruction activity (Keystore=<keystore>, keyName=<*keyname*>, uid=<*target uid*>, requested by <*pkg name*>: uid=<*uid*> role=<*SystemApp, UserApp*> | <*Administrator, NonAdministrator*>) failed with error <*error msg*>	Shows success or failure of deleting keys from the keystore.
Key integrity check failed: key filename=<*filename*>, uid=<*uid*>	Shows the integrity check of a key in the keystore has failed

4.4.5.11 Certification Revocation Events

Message	Description
Admin id <*admin pkg name*> has <*enabled, disabled*> certificate revocation check for <*pkg name*>	Shows enabling or disabling of certificate revocation checking. <*pkg name*> shows if changed for specific packages. "*" is shown for all packages.
Admin id <*admin pkg name*> has <*enabled, disabled*> OCSP for <*pkg name*>	Shows enabling or disabling of OCSP checking. <*pkg name*> shows if changed for specific packages. "*" is shown for all packages.
Installing certificate succeeded. Keystore <*keystore*>, <*certificate information*> Deleting certificate succeeded. Keystore <*keystore*>, <*certificate information*>	Shows the status of adding and removing user certificates into the Trust Anchor Database.
Clearing credentials succeeded. Keystore : <*Wi-Fi, VPN and Apps*> Clearing credentials succeeded. Keystore : Default	Shows clearing of credentials in the Trust Anchor Database that are associated with specific applications (or all).
Chain verification failed. Cert[<*num*>]: <*cert subject*> Issuer: <*cert issuer*> Reason: <*error msg*>	Shows when X.509v3 certificate validation errors occur
CertPathValidator failed: Unable to determine revocation status due to network error	Shows error messages for CRL or OCSP failures

4.4.5.12 Wi-Fi Management Events

Message	Description
Admin id <*admin pkg name*> has changed WiFi SSID restriction to <*true/false*>	Enables or disables SSID white/black listing

Message	Description
Admin id <*admin pkg name*> has <*added/removed*> SSID <*SSID name*> to the restriction <*blacklist/whitelist*>	The admin has added or removed specific SSIDs to the whitelist or blacklist. "*" can be used on the blacklist to note that all networks except those explicitly allowed will be blocked
Admin id <*admin pkg name*> has removed all SSIDs from the restriction <*blacklist/whitelist*>	The admin has removed all SSIDs from the specified list
Admin id <*admin pkg name*> has <*allowed/blocked*> access to WiFi SSID <*SSID name*>	Individual SSID control without enabling white/black listing
Admin id <*admin pkg name*> has set a new wifi profile: SSID: <*SSID name*> Security type <*security*> CA certificate: <*cert*>	The admin has pushed a new Wi-Fi profile to the device

4.4.5.13 Remote Session Events

Message	Description
Application (<*pkg name*>, <*uid*>) has started a SSL/TLS handshake with a remote connection endpoint (<*dst name*>:<*dst port*>) Application (<*pkg name*>, <*uid*>) has finished a SSL/TLS handshake with a remote connection endpoint (<*dst name*>:<*dst port*>) Application (<*pkg name*>, <*uid*>) has finished a SSL/TLS session with a remote connection endpoint (<*dst name*>:<*dst port*>) Application (<*pkg name*>, < *uid*>) has started a HTTPS handshake with a remote connection endpoint (<*dst name*>:<*dst port*>) Application (<*pkg name*>, <*uid*>) has finished a HTTPS handshake with a remote connection endpoint (<*dst name*>:<*dst port*>) Application (<*pkg name*>, <*uid*>) has finished a HTTPS session with a remote connection endpoint (<*dst name*>:<*dst port*>)	Shows applications have started SSL, TLS or HTTPS connections with remote endpoints.
Application (<*pkg name*>, < *uid*>) got SSL protocol exception: *Handshake (failed, aborted)* Cause: <*error msg*>	Shows applications have attempted to start SSL, TLS or HTTPS connections with remote endpoints but there was a failure as specified in the <*error msg*>

Message	Description
Identifier verification failed. Presented identifier: <*identifier*> List of reference identifiers: <*reference identifiers*>	Shows the presented identifier to be checked and rejected
Wi-Fi is failed to connect to <*access point*> network. Reason: <*msg reason*>. Wi-Fi is connected to <*SSID*> network using <*Type of Channel*> channel Wi-Fi is disconnected from <*SSID*> network using EAP-TLS channel	EAP-TLS status messages Errors: • Authentication Failure – invalid client certificate
wpa_supplicant messages: SSL handshake failed: SSL_connect error <*error #*>:SSL routines:<*routine*>:<*error msg*>	Certificate errors are shown in the EAP-TLS processing

4.4.5.14 Application Install/Update Events

Message	Description
Admin <*admin pkg name*> has <*allowed/blocked*> access to Google Playstore (com.android.vending)	Shows when Google Play store is allowed or not
Admin <*admin pkg name*> has <*allowed/disallowed*> installation of non-Google-Play application	Admin has allowed or blocked “Unknown Sources” for installing apps
Starting to install Application <*pkg name*> Install Application <*pkg name*> <*succeeded/failed*>	Installation or update of application has started and succeeded or failed
Uninstall Application <*pkg name*> <*succeeded/failed*>	The removal of the application has succeeded or failed

Message	Description
Admin <*admin pkg name*> has <*installed/removed*> application <*pkg name*>	The admin had installed or removed the application by policy
Application installed from <*path*> insecure file path	The installed application has an insecure file path (such as a bad file extension even though the file was a proper package)
Admin <*admin pkg name*> has installed application from <*path*>	The admin has installed an application from the path specified
Admin <*admin pkg name*> has prevented installation of <*pkg names*>	The policy has been set to prevent the installation of the listed applications. "*" is used to denote all applications should be prevented.
Admin <*uid*> has <*added/removed*> <*signature*> to app signature <*whitelist/blacklist*>	The admin has added or removed a developer signature to an application whitelist or blacklist. This will allow or block all applications signed by this signature.
Admin <*uid*> has <*added/removed*> <*signature*> to package name <*whitelist/blacklist*>	The admin has added or removed a package name to an application whitelist or blacklist. "*" and "?" can be used as wildcards in denoting the package names.

4.4.5.15 Sync Events

Message	Description
Syncing account <*account name*> succeeded	The account has successfully completed a sync to the associated server

4.4.5.16 User Events

Message	Description
screen-lock enabled : password	This shows the user has set or reset their password
The device time has been changed. Current Time = <*time*> [<*uid*>, <*pid*>]	The device time has been changed

4.4.5.17 Location Events

Message	Description
Admin <*admin pkg name*> has <*started/stopped*> GPS	The admin has started or stopped the GPS radio
Admin <*admin pkg name*> has <*enabled/disabled*> location provider <*GPS/network/passive*>	The admin has enabled or disabled the specified location provider service

4.4.5.18 FOTA Update Events

Message	Description
Software update: Software update <*version*> succeeded Software update: Software update <*version*> failed	Shows status of FOTA update process for All carriers except Verizon Wireless. The first message will appear before the reboot where the update will be applied. Success or failure of update will be recorded on the boot after the update process has run.
Software update: Software update <*packageName*> started Software update: Software update <*version*> succeeded Software update: Software update <*version*> failed	Shows status of FOTA update process Verizon Wireless. The first message will appear before the reboot where the update will be applied. Success or failure of update will be recorded on the boot after the update process has run.

4.4.5.19 Integrity Events

Message	Description
Verification failed. Unable to restart you phone. The integrity verification has failed. You need to reset your phone to the factory default settings. This will erase you data.	NOTE: This is shown on the screen and forces a data wipe (so no log).

Message	Description
BoringSSL self-test <*msg*>	Shows successful completion of BoringSSL self-tests

4.5 Secure Delivery

While a Samsung device requires initial configuration before it can be added to the enterprise environment, it is also critical to ensure that the device is received prior to configuration in a secure manner, free from tampering or modification.

It is very important that the devices to be deployed into the enterprise are obtained from reputable carriers to reduce the likelihood that tampering of devices may occur.

Upon receipt, the boxes containing the device should have both a tracking label and two labels placed at either end of the box to indicate whether the box has been opened prior to delivery. If these seals are broken, do not accept the device and return it to your supplier.

The tracking label should look similar to Figure 2 - Tracking label, while the two tamper labels should appear similar to Figure 3 - Security Seal (Black) or Figure 4 - Security Seal (White).

Figure 2 - Tracking label

Figure 3 - Security Seal (Black)

Figure 4 - Security Seal (White)

4.5.1 Evaluation version

There are a number of components to determining the device that is being used and the components on that device (such as the operating system version, the build version, etc.). These are all contained under ***Settings/About device***. The following are version information that can be found:

- ***Model number*** – this is the hardware model (this is carrier specific, so for example a Samsung Galaxy S4 on Verizon Wireless has a different model number than on AT&T)
- ***Android version*** – this is the Android OS version
- ***Build number*** – this is the specific binary image version for the device
- ***Security Software Version*** – this shows the Common Criteria evaluations and the version of the software components related to those evaluations on the device

For the Common Criteria evaluation for the mobile device, this will show:

MDF v**ABC** Release **XYZ**

Where **ABC** is the version of the MDFPP and **XYZ** is the version number of the software that has been validated.

4.5.2 Pre-packaged Software Versions

Samsung Android devices come with large amounts of software apps to provide the full breadth of functionality expected by the customer. Some of the apps come from Google, some from Samsung, and others from the cellular carrier. For a list of the apps and their versions contained on a specific device, visit the website where you can download the CC Mode app and select the device you are using. This will provide a complete list of the software installed on the evaluated device.

4.5.2.1 Software Versions on Device

To verify the versions of any software on the device (compared to the list from the website), open ***Settings/Application manager***. Under the heading ***All***, you will see every application on the device (both those that are pre-installed and any you have installed). Selecting an application will display its properties. The version number is shown at the top under the name.

Note: Using adb (USB debugging must be enabled to use adb) it is possible to extract all package version information at once.

4.6 Secure Updates

Once a device has been deployed, it may be desirable to accept updates to the software on the device to take advantage of the latest and greatest features of Samsung Android. Updates are provided for devices as determined by Samsung and the carriers based on many factors.

When updates are made available, they are signed by Samsung with a private key that is unique to the device/carrier combination (i.e. a Galaxy S7 on Verizon will not have an update signed with the same key as a Galaxy S7 on AT&T). The public key is embedded in the bootloader image, and is used to verify the integrity and validity of the update package.

When updates are made available for a specific device (they are generally rolled out in phases across a carrier network), the user will be prompted to download and install the update (see the User Guide for more information about checking for, downloading and installing the update). The update package is checked automatically for integrity and validity by the software on the device. If the check fails the user is informed that there were errors in the update and the update will not be installed.

4.6.1 Allowed Update Methods

When CC Mode is enabled, only Firmware Over the Air (FOTA) updates are allowed to be installed on the device. Other methods for installing updates (such as ODIN or Samsung KIES) are blocked and cannot

be used to update the firmware. This provides insurance against local, physical attacks that could change the software unknowingly.

4.6.2 Blocking Updates

It is possible to block FOTA updates on a device by setting **allowOTAUpgrade()** to be false via the MDM. This can be used to either freeze the software installed or to allow an organization time to test the update before letting it roll out to the user community.

5 Operational security

5.1 Modes of operation

The TOE can be operated in four different modes, depending on the role of the user accessing the device:

- Administrator mode;
- User mode;
- Error mode; and
- Recovery mode

A device is considered to be in **Administrator mode** before it is delivered to the user. The device is prepared and configured for deployment in the enterprise environment via the Samsung Enterprise SDK. The TOE administrators are trusted to follow and apply all administrator guidance in a trusted manner. An unprivileged user will not have access to this mode of operation.

If an error or operational failure occurs during the transition from Administrator mode (causing the device to momentarily enter the Error mode of operation) to User mode, the administrator should follow the guidance for the Mobile Device Management Solution to rectify the failure and restore the device to normal operational abilities. If it is not possible to adequately eliminate the error or operational failure, the device is not to be delivered to an end user and should be returned to the supplier.

After the device is configured in accordance with the Common Criteria evaluated settings, the device is ready for deployment to a user. When the user receives the device, only the TouchWiz user interface will be visible and no further changes to the security configuration are possible. Once deployed to a user, the device will be operating in **User Mode**. Within User Mode, the only security relevant functions accessible for the user are ‘lock screen password protection’, ‘change of password’ and ‘local device wipe’. Typically, an administrator will not access the device in this mode of operation.

The TOE may also be placed into Recovery mode, bypassing the standard boot process and allowing for configuration changes to be made to the installation of Android. However, this requires the boot loader for the device to be unlocked and is therefore considered out of scope for this environment.

5.2 Wiping data

The evaluated security configurations provide the ability to both locally or remotely wipe data on the device. Based on the device configuration, it is possible to wipe data at the device level, the KNOX container level or both.

An enterprise initiated remote wipe command (either for the device or just the KNOX container, depending on the configuration) occurs under the following conditions:

- The enterprise sends a remote wipe command to the device:
    - when the device has been lost or stolen;
    - in response to a reported incident;
    - in an effort to resolve current mobile issues; and
    - for other procedural reasons such as when an Android device end user leaves the organization.

5.2.1 Wiping the device

The evaluated security configuration provides for a local and a remote wiping process of Android user devices. This type of wipe works at the storage level and will wipe all data on the device. In a KNOX container configuration this will wipe all data including the KNOX container (as well as everything not in the container). This type of wipe is available in all configurations.

The local wipe is manually initiated by the Android device user or after an exceeded number of incorrect login attempts. The remote wipe process is in general remotely initiated by the Enterprise and Mobile Device Administrator via a remote wipe command.

5.2.2 Wiping the KNOX Container

When a KNOX container has been enabled it is also possible to wipe only the data stored in the KNOX container. A wipe of the container data will remove the container, including apps and data, but it will not remove anything outside the KNOX container. This process must be initiated remotely by the Enterprise and Mobile Device Administrator via a remote wipe container command.

The only way for a user to locally wipe the KNOX container is to unenroll the device from the control of the MDM. When this is done the KNOX container, all data and apps as well as the MDM Agent will all be removed from the device.

5.3 VPN Client Use

While Samsung devices come with a Common Criteria-certified VPN client, some Enterprise customers may have a need for a VPN client from a third party vendor. Android provides the public class android.net.VpnService for third party vendors to build VPN clients that can be installed and used for functionality beyond that which is provided by the Android or Samsung built-in VPN clients. VPN client software built using this interface may provide their own management interface outside of that provided by Samsung.

More information about this can be found here: https://developer.android.com/reference/android/net/VpnService.html.

5.4 Additional notes on operational security

Common Criteria Part 3 does require operational user guidance for the following:

- User-accessible functions and privileges that should be controlled in a secure processing environment, including appropriate warnings.
- Secure usage of available interfaces.
- Security parameters of interfaces and functions under the control of the user and their secure values.
- Each type of security-relevant event relative to the user-accessible functions.

Administrators and users are considered to use a Samsung Enterprise device. As described in previous sections of this document, the administrator is responsible for configuration and installation of the device. The end user receives the device in an operational state where no further security configuration is possible. The only user accessible user functions are ‘lock screen password protection’, ‘change of password’ and ‘local device wipe’.

The user is responsible to obey the provided user guidance and to not actively working against the protection of the device data.

The TOE Administrators are trusted to follow and apply all administrator guidance, resp. [MDMG] in a trusted manner. [MDMG] provides further operational user guidance.