Test DPC User Guide for Independent App Developers & Software Vendors

Version 1.0

Introduction

To simplify the process of testing Enterprise functionality in your app, we've released and open-sourced a sample device policy controller called Test DPC. When developing functionality or apps for use in businesses, you can use Test DPC to simulate the many features of Android for Work. Test DPC simplifies testing and development because you can use it to set the kinds of policies an IT Administrator might enforce on an Android device that's used in the workplace. You can set app and intent restrictions, set up managed work profiles, enforce policies, and can even set up fully managed Android devices—something you might find in a public place.

We've updated Test DPC with new functionality for the Marshmallow (M) release (Android 6.0/API level 23). To find out more about Android for Work visit developer.android.com/work.

Note: This guide focuses on independent app developers and software vendors. There are many features that are not explained in this guide, because they don't apply to our target audience. If you are an enterprise mobility management (EMM) company, or are interested in building your own device policy controller to manage Android for Work enabled devices, learn more about Building a Device Policy Controller.

Glossary

COSU—corporate-owned, single use. An implementation of the device owner configuration, where the device is only intended for a specific use case and, thus, only a single app or small set of apps is accessible on the device, perhaps even without access to the Android launcher. M introduces enhancements to the device-owner APIs to enable this sort of configuration.

DPC—Device policy controller; an EMM management app that can be configured as one of two modes of a device administrator—profile owner or device owner. The DPC can execute a variety of policy controls, depending on its mode of configuration.

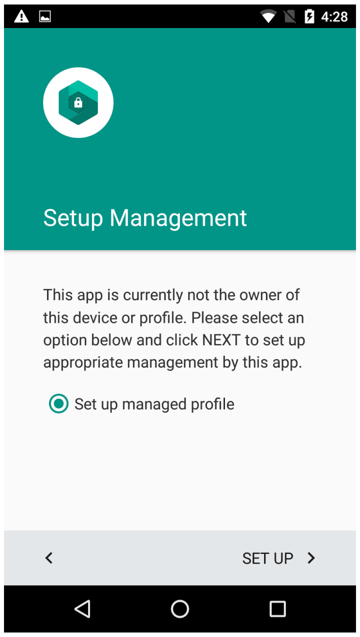
device owner—A single-user Android for Work configuration, wherein the primary user profile, and thus the device itself, is managed by a DPC configured as a device owner. The management capabilities of the DPC are applicable to the entire device, and thus this mode of use is typically associated with corporate liable or line-of-business devices. Also known as **managed device** or **corporate liable.**

profile owner—A multiuser Android for Work configuration, wherein a separate managed profile is provisioned alongside an unmanaged primary profile (which is traditionally designated for personal, unmanaged data). In this case, the DPC becomes the profile owner of the managed profile, and its management capabilities are restricted to just that profile and don't extend into the primary profile. This mode of use is typically associated with a BYOD mobility strategy. Also known as **bring your own device (BYOD).**

work profile—A secondary profile, managed by a DPC, configured as a profile owner. Because it's provisioned alongside an unmanaged primary profile, a managed profile shares elements, such as a launcher, a unified settings app, and even app APKs with the primary profile. There is, however, strict data separation between the managed profile and the primary profile. Also known as **managed profile.**

Test applications

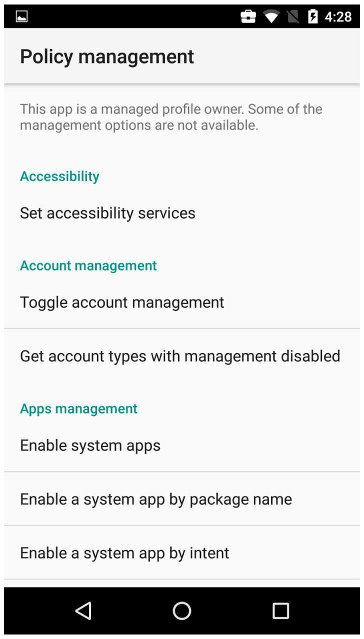
Test DPC app


4:28

Setup Management

This app is currently not the owner of this device or profile. Please select an option below and click NEXT to set up appropriate management by this app.

[✓ Set up managed profile]


4:28

Policy management

This app is a managed profile owner. Some of the management options are not available.

Accessibility

Set accessibility services

Account management

Toggle account management

Get account types with management disabled

Apps management

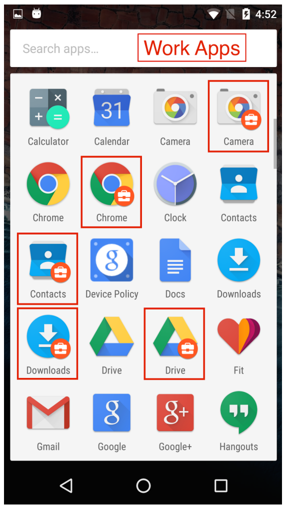
Enable system apps

Enable a system app by package name

Enable a system app by intent

The Test DPC app can be found on Google Play, and the source code for the app can be found on GitHub. When using this app, please refer to the following instructions for inflating a **work profile**:

1. After installing Test DPC, launch it and click **Set up managed profile**.
2. When prompted by the system, click **Set up**, ensuring Test DPC's logo is highlighted on the screen.
3. If your device is not encrypted, do so when prompted, and follow the briefcase notification after reboot to continue provisioning.
4. Once provisioning is complete, you'll be prompted to add an account. Feel free to do so, or you can add an account later.
5. Verify that there are now work apps in your app drawer. The work apps are badged with the orange suitcase icon .



NFC Provisioning app

The NFC Provisioning app is used for device owner provisioning. To verify your app for compatibility with a device owner or COSU, you need Test DPC and the NFC Provisioning app to get started. The source for the NFC Provisioning app can be found on GitHub. When using this app, please refer to the following instructions to provision the device and set Test DPC as **device owner**:

1. Create a text file called `nfcprovisioning.txt` with the following content:

   Android 5.0 or 5.1 Lollipop

   ```
   android.app.extra.PROVISIONING_DEVICE_ADMIN_PACKAGE_NAME=com.afwsamples.
   testdpc
   android.app.extra.PROVISIONING_DEVICE_ADMIN_PACKAGE_DOWNLOAD_LOCATION=ht
   tps://sites.google.com/site/afwhost/home/testdpc/comafwsamplesTestDPC.ap
   k
   android.app.extra.PROVISIONING_DEVICE_ADMIN_PACKAGE_CHECKSUM=uOe3bRbdoml
   qbnvNJ7nnSNVvBBE=
   android.app.extra.PROVISIONING_LOCALE=en_US
   android.app.extra.PROVISIONING_TIME_ZONE=America/New_York
   android.app.extra.PROVISIONING_LEAVE_ALL_SYSTEM_APPS_ENABLED
   ```

   Android 6.0 Marshmallow

   ```
   android.app.extra.PROVISIONING_DEVICE_ADMIN_PACKAGE_NAME=com.afwsamples.
   testdpc
   android.app.extra.PROVISIONING_DEVICE_ADMIN_PACKAGE_DOWNLOAD_LOCATION=ht
   tps://sites.google.com/site/afwhost/home/testdpc/comafwsamplesTestDPC.ap
   k
   android.app.extra.PROVISIONING_DEVICE_ADMIN_SIGNATURE_CHECKSUM=gJD2YwtOi
   WJHkSMkkIfLRlj-quNqG1fb6v100QmzM9w=
   # note: checksum must be URL-safe
   android.app.extra.PROVISIONING_LOCALE=en_US
   android.app.extra.PROVISIONING_TIME_ZONE=America/New_York
   ```

2. Push that `nfcprovisioning.txt` text file to your programmer device:

   ```
   adb push <path-to-nfcprovisioning.txt> /sdcard/
   ```

3. Open the NFC Provisioning app and ensure that `com.afwsamples.testdpc` is auto-populated.
4. Bump the devices and touch to beam.
5. Follow onscreen instructions on the target device.

Test DPC features

The following table lists Test DPC features, sorted by their location within Test DPC.

Test DPC Feature Name	API Level			DO[1]	PO[1]	Relevant Android for Work Features
	21	22	23			
Accessibility						
Set accessibility services	✓	✓	✓	✓	✓	Accessibility services
Account management						
Toggle account management	✓	✓	✓	✓	✓	Not relevant in this guide
Get account types with management disabled	✓	✓	✓	✓	✓	Not relevant in this guide
Apps management						
Enable system apps	✓	✓	✓	✓	✓	Not relevant in this guide
Enable a system app by package name	✓	✓	✓	✓	✓	Not relevant in this guide
Enable a system app by intent	✓	✓	✓	✓	✓	Not relevant in this guide
Hide apps	✓	✓	✓	✓	✓	Not relevant in this guide
Unhide apps	✓	✓	✓	✓	✓	Not relevant in this guide
Manage app restrictions	✓	✓	✓	✓	✓	Set app restrictions
Block uninstallation						
Block uninstallation by package name	✓	✓	✓	✓	✓	Not relevant in this guide
Block uninstallation list	✓	✓	✓	✓	✓	Not relevant in this guide
Camera and screen capture						
Disable camera	✓	✓	✓	✓	✓	Not relevant in this guide
Capture image	✓	✓	✓	✓	✓	Not relevant in this guide
Capture video	✓	✓	✓	✓	✓	Not relevant in this guide
Disable screen capture	✓	✓	✓	✓	✓	Not relevant in this guide

[1] DO = device owner; PO = policy owner

Certificate management						
Install a private key certificate	✓	✓	✓	✓	✓	Delegated certificate installation
Override client certificates with alias			✓	✓	✓	Silent enterprise certificate access
Install a CA[2] certificate	✓	✓	✓	✓	✓	Delegated certificate installation
Get a list of CA certificates	✓	✓	✓	✓	✓	Delegated certificate installation
Remove all CA certificates	✓	✓	✓	✓	✓	Delegated certificate installation
Manage delegated certificate installer			✓	✓	✓	Delegated certificate installation
Data usage						
Network data usage stats			✓	✓	✓	Data usage tracking
Wifi management						
Create Wi-Fi configuration	✓	✓	✓	✓	✓	Not relevant in this guide
DO created Wi-Fi configs are modifiable				✓		Not relevant in this guide
Modify Wi-Fi configuration	✓	✓	✓	✓	✓	Not relevant in this guide
Input methods						
Set input methods	✓	✓	✓	✓	✓	Whitelist input methods
Keyguard features						
Disable widgets				✓		Not relevant in this guide
Disable secure camera				✓		Not relevant in this guide
Disable secure notifications				✓		Profile owner keyguard management
Disable unredacted notifications			✓	✓	✓	Profile owner keyguard management
Disable trust agents			✓	✓	✓	Not relevant in this guide
Disable fingerprint			✓	✓	✓	Not relevant in this guide
Lock screen						
Max time to screen lock (seconds)	✓	✓	✓	✓	✓	Not relevant in this guide
Max password failures for local wipe	✓	✓	✓	✓	✓	Not relevant in this guide

[2] certificate authority (CA)

Lock task						
Manage lock task list				✓		COSU EMM controls
Check lock task permitted				✓		COSU EMM controls
Start lock task mode				✓		COSU EMM controls
Stop lock task mode				✓		COSU EMM controls
Managed profile policy						
- Manage work profile specific policy	✓	✓	✓		✓	- *See below*
- Cross-profile intents						
- Add cross-profile intents	✓	✓	✓		✓	Cross-profile sharing
- Clear all cross-profile intents	✓	✓	✓		✓	Cross-profile sharing
- Cross-profile widget providers						
- Enable app widgets in a work profile app	✓	✓	✓		✓	Cross-profile app widgets
- Remove app widgets in a work profile app	✓	✓	✓		✓	Cross-profile app widgets
- Policies						
- Disable Bluetooth contact sharing			✓		✓	Work contacts in personal contexts
- Disable cross-profile caller ID	✓	✓	✓		✓	Work contacts in personal contexts
- Disallow cross-profile copy/paste	✓	✓	✓		✓	Cross-profile copy and paste
- Work profile management						
- Removing work profile	✓	✓	✓		✓	*Self explanatory*
Permission Management						
Get/Set default permission policy			✓	✓	✓	Policies for runtime permissions
Manage app permissions			✓	✓	✓	Policies for runtime permissions
Single-use devices						
Disable status bar				✓		COSU EMM controls
Re-enable status bar				✓		COSU EMM controls
Disable keyguard				✓		COSU EMM controls

Re-enable keyguard				✓		COSU EMM controls
Start kiosk mode				✓		COSU EMM controls
System update						
Manage system update policy				✓		Not relevant in this guide
User management						
Create and initialize user				✓		Not relevant in this guide
Remove user				✓		Not relevant in this guide
User restrictions						
Set user restrictions						
- Allow web links to apps of the parent			✓	✓	✓	Cross-profile app linking
- Disallow add user				✓		Not relevant in this guide
- Disallow adjust volume				✓		Not relevant in this guide
- Disallow apps control	✓	✓	✓	✓	✓	Not relevant in this guide
- Disallow config Bluetooth				✓		Not relevant in this guide
- Disallow config cell broadcasts				✓		Not relevant in this guide
- Disallow config credentials	✓	✓	✓	✓	✓	Not relevant in this guide
- Disallow config mobile networks				✓		Not relevant in this guide
- Disallow config tethering				✓		Not relevant in this guide
- Disallow config VPN	✓	✓	✓	✓	✓	Not relevant in this guide
- Disallow config Wi-Fi				✓		Not relevant in this guide
- Disallow create windows				✓		Not relevant in this guide
- Disallow cross-profile copy/paste	✓	✓	✓	✓	✓	Cross-profile copy and paste
- Disallow debugging features	✓	✓	✓	✓	✓	Not relevant in this guide
- Disallow factory reset				✓		Not relevant in this guide
- Disallow fun				✓		Not relevant in this guide
- Disallow install apps	✓	✓	✓	✓	✓	Not relevant in this guide
- Disallow install unknown sources	✓	✓	✓	✓	✓	Not relevant in this guide

- Disallow modify accounts	✓	✓	✓	✓	✓	Not relevant in this guide
- Disallow mount physical media				✓		Not relevant in this guide
- Disallow network reset				✓		Not relevant in this guide
- Disallow outgoing beam	✓	✓	✓	✓	✓	NFC sharing from the managed profile
- Disallow outgoing calls				✓		Not relevant in this guide
- Disallow remove user				✓		Not relevant in this guide
- Disallow safe boot				✓		COSU EMM controls
- Disallow share location	✓	✓	✓	✓	✓	Show location access status for managed profiles in Settings > Location
- Disallow SMS				✓		Not relevant in this guide
- Disallow uninstall apps	✓	✓	✓	✓	✓	Not relevant in this guide
- Disallow unmute microphone				✓		Not relevant in this guide
- Disallow USB file transfer				✓		Not relevant in this guide
- Ensure verify apps	✓	✓	✓	✓	✓	Not relevant in this guide
Settings management						
Keep the device on while plugged in				✓		COSU EMM controls
Allow installs from unknown sources	✓	✓	✓	✓	✓	Not relevant in this guide
Device owner management						
Wipe data				✓		Not relevant in this guide
Remove this device owner				✓		*Self explanatory*

Android for Work features

The Android for Work features described in this section are sorted by supported API levels:

- Features supported in API level 21 (Android 5.0 Lollipop)
- Features supported in API level 22 (Android 5.1 Lollipop)
- Features supported in API level 23 (Android 6.0 Marshmallow)

Features supported in API level 21 (Android 5.0 Lollipop)

The following features are supported in API level 21 (Android 5.0 Lollipop):

- Cross-profile camera/multimedia capture using first-party apps
- Cross-profile app widgets
- Whitelist input methods
- Intent forwarding
- Cross-profile sharing
- Set app restrictions
- Accessibility services
- Cross-profile copy and paste

Cross-profile camera/multimedia capture using first-party apps

This feature ensures that apps within the managed profile are able to utilize first-party apps when capturing multimedia. Examples include taking a photo, shooting video, or recording audio.

Test environment—Device with an inflated work profile and a camera app installed in the personal profile.

Test scenarios—Use an app in the managed profile to test these functions:

1. Record an audio and ensure the audio is properly recorded and usable.
2. Capture a photo and ensure the audio is properly taken and usable.
3. Record a video and ensure the video is properly recorded and usable.

Cross-profile app widgets

This feature allows a managed profile administrator to whitelist some apps to publish widgets on the home screen. By default, no apps are whitelisted.

Test environment—Device with an inflated work profile, and an app installed in the work profile that has a widget.

Test scenarios

1. Long-press on the Home screen, and click **Widgets**.

2. Browse the widget catalog and ensure a badged widget for the app is not present.
3. Using Test DPC, click **Manage work profile specific policy** > **Enable app widgets in a work profile app**.
4. Enable one app from the list of work profile apps with widgets.
5. Long-press on the Home screen, and click **Widgets**.
6. Browse the widget catalog and ensure a badged widget for the app is available.
7. Add that widget to the Home screen.
8. Ensure the widget content is relevant to only work profile.
9. Reboot the device, and ensure the widget is present after reboot and still displays accurate content.
10. Using Test DPC, click **Manage work profile specific policy** > **Remove app widgets in a work profile app**, and select the app you tested.
11. Ensure the widgets from the relevant have been removed from the launcher and can't be selected from the widget catalog.

Whitelist input methods

Input Method Editor (IME) selection allows the DPC to define what IMEs are permitted by setting a whitelist; however, system IMEs are always available to the user. IMEs apply to both profiles; there are no work profile-specific IMEs.

Test environment—Device with an inflated work profile, and non-system IMEs installed on the personal side.

Test scenarios:

1. Navigate to **Settings** > **Language & input** > **Current keyboard** > **Choose keyboards**, and switch from the default to a non-system IME.
2. Navigate to **Settings** > **Language & input** > **Current keyboard** > **Choose keyboards**, and switch to a system IME.
3. Using Test DPC, click **Set input methods**, deselect every non-system IME, and click **OK**.
4. System IMEs should be enabled and greyed out, preventing you from disabling them.
5. Navigate to **Settings** > **Language & input** > **Current keyboard** > **Choose keyboards**, and ensure the non-system IMEs are disabled and can't be enabled.

Intent forwarding

Depending on the type of intent, the managed profile may resolve that intent or forward it to the primary profile.

Test environment—Device with an inflated work profile.

Test scenario of a sample intent filter:

1. Using Test DPC, navigate to **Manage work profile specific policy** > **Add cross-profile intents**.
    b. If you want the intent to be forwarded from the work profile to the personal profile, select **FLAG_MANAGED_CAN_ACCESS_PARENT**.
    c. If you want the intent to be forwarded the other way, select **FLAG_PARENT_CAN_ACCESS_MANAGED**.

2. Choose values for actions, categories, schemes, and data types by selecting options from the drop-down list and clicking +.
3. When you're finished selecting the necessary options, create the cross-profile intent filter by clicking **Add**.
4. Check if your intent if forwarded properly. For example:
    a. Create an intent filter with flag `FLAG_MANAGED_CAN_ACCESS_PARENT`, action `android.intent.action.VIEW`, category `android.intent.category.DEFAULT`, and scheme `http.`
    b. Fire an intent to launch a browser using adb:
    ```
    adb shell am start -a android.intent.action.VIEW -d
    http://www.google.com --user <USER_ID_OF_MANGED_PROFILE>
    ```
    Note: The userID of the managed profile can also be identified via adb:
    ```
    adb shell pm list users
    ```
    c. If a browser is installed in the work profile, ensure you're prompted to open with the work browser or a personal browser via the personal side button; otherwise, if no browser is installed in the work profile, ensure the personal side browser directly resolves the intent.

Refer to the following table for expected resolution of intents fired from the work profile:

Action	Expected behavior	Shell command
ACTION_SET_ALARM	Should invoke handlers within managed profile (if present); otherwise, forward to primary user.	`adb shell am start -a android.intent.action.SET_ALARM --ei android.intent.extra.alarm.HOUR 13 --ei android.intent.extra.alarm.MINUTES 37 --user <USER_ID>`
ACTION_SET_TIMER	Should invoke handlers within managed profile (if present); otherwise, forward to primary user.	`adb shell am start -a android.intent.action.SET_TIMER --ei android.intent.extra.alarm.LENGTH 120 --user <USER_ID>`
ACTION_SHOW_ALARMS	Should invoke handlers within managed profile (if present); otherwise, forward to primary user.	`adb shell am start -a android.intent.action.SHOW_ALARMS --user <USER_ID>`
ACTION_INSERT (Browser, add a bookmark)	Should invoke handlers within managed profile (if present).	`adb shell am start -a android.intent.action.INSERT -d content://browser/bookmarks -e title Example -e url "http://www.example.com" --user <USER_ID>`
ACTION_VIEW (Browser, load a URL)	Should invoke handlers within managed profile (if present).	`adb shell am start -a android.intent.action.VIEW -c android.intent.category.BROWSABLE -d "http://www.example.com" --user <USER_ID>`
ACTION_INSERT (Calendar, add an event)	Should invoke handlers within managed profile (if present).	
ACTION_IMAGE_CAPTURE	Should invoke handlers within managed profile (if present); otherwise, forward to primary user.	`adb shell am start -a android.media.action.IMAGE_CAPTURE --user <USER_ID>`

ACTION_VIDEO_CAPTURE	Should invoke handlers within managed profile (if present); otherwise, forward to primary user.	`adb shell am start -a android.media.action.VIDEO_CAPTURE --user <USER_ID>`
INTENT_ACTION_STILL_IMAGE_CAMERA	Should invoke handlers within managed profile (if present); otherwise, forward to primary user.	`adb shell am start -a android.media.action.STILL_IMAGE_CAMERA --user <USER_ID>`
INTENT_ACTION_VIDEO_CAMERA	Should invoke handlers within managed profile (if present); otherwise, forward to primary user.	`adb shell am start -a android.media.action.VIDEO_CAMERA --user <USER_ID>`
ACTION_SENDTO (Email, compose a message)	Should invoke handlers within managed profile (if present).	`adb shell am start -a android.intent.action.SENDTO -d "mailto:user@example.com" --user <USER_ID>`
ACTION_VIEW (Email, compose a message)	Should invoke handlers within managed profile (if present).	`adb shell am start -a android.intent.action.VIEW -c android.intent.category.BROWSABLE -d "mailto:user@example.com" --user <USER_ID>`
ACTION_VIEW_DOWNLOADS	Should open the instance of the 'Downloads' app belonging to the user specified in the intent.	`adb shell am start -a android.intent.action.VIEW_DOWNLOADS --user <USER_ID>`
ACTION_GET_CONTENT	Should invoke handlers within managed profile (if present); otherwise, forward to primary user (if allowed by DPC).	
ACTION_OPEN_DOCUMENT	Should invoke handlers within managed profile (if present); otherwise, forward to primary user (if allowed by DPC).	
ACTION_VIEW (Maps, show a location)	Should invoke handlers within managed profile (if present).	`adb shell am start -a android.intent.action.VIEW -d "geo:0,0?q=Buckingham Palace" --user <USER_ID>`
ACTION_DISPLAY_AUDIO_EFFECT_CONTROL_PANEL	Should open the audio effect control panel for the user specified in the intent.	
ACTION_VIEW (Media, play a video file)	Should invoke handlers within managed profile (if present).	`adb shell am start -a android.intent.action.VIEW -d "http://vjs.zencdn.net/v/oceans.mp4" -t "video/mp4" --user <USER_ID>`
ACTION_VIEW (Media, play an audio file)	Should invoke handlers within managed profile (if present).	`adb shell am start -a android.intent.action.VIEW -d "http://www.w3schools.com/html/horse.mp3" -t "audio/*" --user <USER_ID>`
INTENT_ACTION_MEDIA_PLAY_FROM_SEARCH	Should invoke handlers within managed profile (if present).	`adb shell am start -a android.media.action.MEDIA_PLAY_FROM_SEARCH -e android.intent.extra.focus "vnd.android.cursor.item/*" --user <USER_ID>`
ACTION_IMAGE_CAPTURE_SECURE	Should invoke handlers within managed profile (if present); otherwise, forward to primary user.	`adb shell am start -a android.media.action.IMAGE_CAPTURE_SECURE --user <USER_ID>`
INTENT_ACTION_STILL_IMAGE_CAM	Should invoke handlers within	`adb shell am start -a`

ERA_SECURE	managed profile (if present); otherwise, forward to primary user.	`android.media.action.STILL_IMAGE_CAMERA_SECURE --user <USER_ID>`
android.provider.MediaStore.RECORD_SOUND	Should invoke handlers within managed profile (if present).	`adb shell am start -a android.provider.MediaStore.RECORD_SOUND --user <USER_ID>`
ACTION_SENDTO (Messaging, compose an sms with sms: URI)	Forward to primary user.	`adb shell am start -a android.intent.action.SENDTO -d "sms:07700900100" -e sms_body "Hello world" --user <USER_ID>`
ACTION_SENDTO (Messaging, compose an sms with smsto: URI)	Forward to primary user.	`adb shell am start -a android.intent.action.SENDTO -d "smsto:07700900100" -e sms_body "Hello world" --user <USER_ID>`
ACTION_SENDTO (Messaging, compose an mms with mms: URI)	Forward to primary user.	`adb shell am start -a android.intent.action.SENDTO -d "mms:07700900100" -e sms_body "Hello world" -e subject "Example" --user <USER_ID>`
ACTION_SENDTO (Messaging, compose an mms with mmsto: URI)	Forward to primary user.	`adb shell am start -a android.intent.action.SENDTO -d "mmsto:07700900100" -e sms_body "Hello world" -e subject "Example" --user <USER_ID>`
ACTION_VIEW (Messaging, compose an sms with sms: URI)	Forward to primary user.	`adb shell am start -a android.intent.action.VIEW -c android.intent.category.BROWSABLE -d "sms:07700900100?body=Hello%20world" --user <USER_ID>`
ACTION_VIEW (Messaging, compose an sms with smsto: URI)	Forward to primary user.	`adb shell am start -a android.intent.action.VIEW -c android.intent.category.BROWSABLE -d "smsto:07700900100?body=Hello%20world" --user <USER_ID>`
ACTION_VIEW (Messaging, compose an mms with mms: URI)	Forward to primary user.	`adb shell am start -a android.intent.action.VIEW -c android.intent.category.BROWSABLE -d "mms:07700900100?body=Hello%20world" --user <USER_ID>`
ACTION_VIEW (Messaging, compose an mms with mmsto: URI)	Forward to primary user.	`adb shell am start -a android.intent.action.VIEW -c android.intent.category.BROWSABLE -d "mmsto:07700900100?body=Hello%20world" --user <USER_ID>`
android.speech.action.RECOGNIZE_SPEECH	Should invoke handlers within managed profile (if present); otherwise, forward to primary user.	
ACTION_VIEW (Play, link to product details page)	Should invoke the work instance of Play Store.	`adb shell am start -a android.intent.action.VIEW -c android.intent.category.BROWSABLE -d market://details?id=com.android.chrome --user <USER_ID>`
ACTION_WEB_SEARCH	Should invoke handlers within managed profile (if present).	`adb shell am start -a android.intent.action.WEB_SEARCH -e query example --user <USER_ID>`

ACTION_ACCESSIBILITY_SETTINGS	Forward to primary user.	`adb shell am start -a android.settings.ACCESSIBILITY_SETTINGS --user <USER_ID>`
ACTION_ADD_ACCOUNT	Forward to primary user.	`adb shell am start -a android.settings.ADD_ACCOUNT_SETTINGS --user <USER_ID>`
ACTION_AIRPLANE_MODE_SETTINGS	Settings are system-level, intent should be forwarded to primary user.	`adb shell am start -a android.settings.AIRPLANE_MODE_SETTINGS --user <USER_ID>`
ACTION_APN_SETTINGS	Settings are system-level, intent should be forwarded to primary user.	`adb shell am start -a android.settings.APN_SETTINGS --user <USER_ID>`
ACTION_APPLICATION_DETAILS_SETTINGS	Should show the app info for the instance of the app belonging to the user specified by the intent.	`adb shell am start -a android.settings.APPLICATION_DETAILS_SETTINGS -d package:com.android.chrome --user <USER_ID>`
ACTION_APPLICATION_DEVELOPMENT_SETTINGS	Settings are system-level, intent should be forwarded to primary user.	`adb shell am start -a android.settings.APPLICATION_DEVELOPMENT_SETTINGS --user <USER_ID>`
ACTION_APPLICATION_SETTINGS	Should show apps installed within the user specified by the intent.	`adb shell am start -a android.settings.APPLICATION_SETTINGS --user <USER_ID>`
ACTION_CAPTIONING_SETTINGS	Settings are system-level, intent should be forwarded to primary user.	`adb shell am start -a android.settings.CAPTIONING_SETTINGS --user <USER_ID>`
ACTION_DATA_ROAMING_SETTINGS	Settings are system-level, intent should be forwarded to primary user.	`adb shell am start -a android.settings.DATA_ROAMING_SETTINGS --user <USER_ID>`
ACTION_DATE_SETTINGS	Settings are system-level, intent should be forwarded to primary user.	`adb shell am start -a android.settings.DATE_SETTINGS --user <USER_ID>`
ACTION_DEVICE_INFO_SETTINGS	Settings are system-level, intent should be forwarded to primary user.	`adb shell am start -a android.settings.DEVICE_INFO_SETTINGS --user <USER_ID>`
ACTION_DISPLAY_SETTINGS	Intent should be forwarded to primary user instance of the Settings app. ACTION_SET_WALLPAPER handlers, live wallpapers, and Daydreams installed in the managed profile should *not* be listed.	`adb shell am start -a android.settings.DISPLAY_SETTINGS --user <USER_ID>`
ACTION_DREAM_SETTINGS	Intent should be forwarded to primary user instance of the Settings app. Daydreams installed in the managed profile should *not* be listed.	`adb shell am start -a android.settings.DREAM_SETTINGS --user <USER_ID>`
ACTION_INPUT_METHOD_SETTINGS	Only input methods on the primary user should be shown. The intent will be forwarded.	`adb shell am start -a android.settings.INPUT_METHOD_SETTINGS --user <USER_ID>`
ACTION_INPUT_METHOD_SUBTYPE_SETTINGS	Only input methods on the primary user should be shown. The intent will be forwarded.	`adb shell am start -a android.settings.INPUT_METHOD_SUBTYPE_SETTINGS --user <USER_ID>`

ACTION_INTERNAL_STORAGE_SETTINGS	Should list usage for both primary user and managed profile.	`adb shell am start -a android.settings.INTERNAL_STORAGE_SETTINGS --user <USER_ID>`
ACTION_LOCALE_SETTINGS	Settings are system-level. Intent should be forwarded to primary user instance of the Settings app.	`adb shell am start -a android.settings.LOCALE_SETTINGS --user <USER_ID>`
ACTION_LOCATION_SOURCE_SETTINGS	Should list usage for both primary user and managed profile.	`adb shell am start -a android.settings.LOCATION_SOURCE_SETTINGS --user <USER_ID>`
ACTION_MANAGE_ALL_APPLICATIONS_SETTINGS	Should show a list of the apps installed by the user specified in the intent.	`adb shell am start -a android.settings.MANAGE_ALL_APPLICATIONS_SETTINGS --user <USER_ID>`
ACTION_MANAGE_APPLICATIONS_SETTINGS	Should show a list of the apps installed by the user specified in the intent.	`adb shell am start -a android.settings.MANAGE_APPLICATIONS_SETTINGS --user <USER_ID>`
ACTION_MANAGE_NETWORK_USAGE	Should open the network usage settings activity within the instance of the app belonging to the user specified in the intent.	`adb shell am start -a android.intent.action.MANAGE_NETWORK_USAGE --user <USER_ID> com.google.android.gm`
ACTION_MEMORY_CARD_SETTINGS	Should list usage for both primary user and managed profile.	`adb shell am start -a android.settings.MEMORY_CARD_SETTINGS --user <USER_ID>`
ACTION_NETWORK_OPERATOR_SETTINGS	Settings are system-level. Intent should be forwarded to primary user instance of the Settings app.	`adb shell am start -a android.settings.NETWORK_OPERATOR_SETTINGS --user <USER_ID>`
ACTION_NFC_PAYMENT_SETTINGS	Should list HCE apps from both primary user and managed profile.	`adb shell am start -a android.settings.NFC_PAYMENT_SETTINGS --user <USER_ID>`
ACTION_NFC_SETTINGS	Settings are system-level. Intent should be forwarded to primary user instance of the Settings app.	`adb shell am start -a android.settings.NFC_SETTINGS --user <USER_ID>`
ACTION_NFCSHARING_SETTINGS	Settings are system-level. Intent should be forwarded to primary user instance of the Settings app.	`adb shell am start -a android.settings.NFCSHARING_SETTINGS --user <USER_ID>`
ACTION_OTHER_SOUND_SETTINGS	Intent is resolved in managed profile instance of the Settings app.	`adb shell am start -a android.settings.ACTION_OTHER_SOUND_SETTINGS --user <USER_ID>`
ACTION_PRINT_SETTINGS	Should show print settings screen for the user specified in the intent.	`adb shell am start -a android.settings.ACTION_PRINT_SETTINGS --user <USER_ID>`
ACTION_PRIVACY_SETTINGS	Settings are system-level. Intent should be forwarded to primary user instance of the Settings app.	`adb shell am start -a android.settings.PRIVACY_SETTINGS --user <USER_ID>`
ACTION_SEARCH_SETTINGS	Should fail to resolve for managed profile user.	`adb shell am start -a android.search.action.SEARCH_SETTINGS --user <USER_ID>`
ACTION_SETTINGS	Should launch the primary user (and only) instance of the Settings app.	`adb shell am start -a android.settings.SETTINGS --user <USER_ID>`
ACTION_SOUND_SETTINGS	Forward to primary user.	`adb shell am start -a android.settings.SOUND_SETTINGS --user <USER_ID>`

ACTION_SYNC_SETTINGS	Forward to primary user.	`adb shell am start -a android.settings.SYNC_SETTINGS --user <USER_ID>`
ACTION_WIFI_IP_SETTINGS	Intent is resolved in managed profile instance of the Settings app.	`adb shell am start -a android.settings.WIFI_IP_SETTINGS --user <USER_ID>`
ACTION_WIFI_SETTINGS	Intent is resolved in managed profile instance of the Settings app.	`adb shell am start -a android.settings.WIFI_SETTINGS --user <USER_ID>`
ACTION_WIRELESS_SETTINGS	Forward to primary user.	`adb shell am start -a android.settings.WIRELESS_SETTINGS --user <USER_ID>`
android.app.action.SET_NEW_PASSWORD	Forward to primary user.	`adb shell am start -a android.app.action.SET_NEW_PASSWORD --user <USER_ID>`
android.net.vpn.SETTINGS	Forward to primary user.	`adb shell am start -a android.net.vpn.SETTINGS --user <USER_ID>`
android.nfc.cardemulation.action.ACTION_CHANGE_DEFAULT	Forward to primary user.	`adb shell am start -a android.nfc.cardemulation.action.ACTION_CHANGE_DEFAULT --user <USER_ID>`
android.settings.ACCOUNT_SYNC_SETTINGS	Forward to primary user.	`adb shell am start -a android.settings.ACCOUNT_SYNC_SETTINGS --user <USER_ID>`
android.settings.BATTERY_SAVER_SETTINGS	Forward to primary user.	`adb shell am start -a android.settings.BATTERY_SAVER_SETTINGS --user <USER_ID>`
android.settings.HOME_SETTINGS	Forward to primary user.	`adb shell am start -a android.settings.HOME_SETTINGS --user <USER_ID>`
android.settings.LICENSE	Forward to primary user.	`adb shell am start -a android.settings.LICENSE --user <USER_ID>`
android.settings.NOTIFICATION_SETTINGS	Forward to primary user.	`adb shell am start -a android.settings.NOTIFICATION_SETTINGS --user <USER_ID>`
android.settings.SHOW_INPUT_METHOD_PICKER	Intent is resolved in managed profile instance of the Settings app.	`adb shell am broadcast -a android.settings.SHOW_INPUT_METHOD_PICKER --user <USER_ID>`
android.settings.SHOW_REGULATORY_INFO	Forward to primary user.	`adb shell am start -a android.settings.SHOW_REGULATORY_INFO --user <USER_ID>`
android.settings.USER_SETTINGS	Forward to primary user.	`adb shell am start -a android.settings.USER_SETTINGS --user <USER_ID>`
android.settings.ZEN_MODE_SETTINGS	Forward to primary user.	`adb shell am start -a android.settings.ZEN_MODE_SETTINGS --user <USER_ID>`
com.android.settings.ACCESSIBILITY_COLOR_SPACE_SETTINGS	Forward to primary user.	`adb shell am start -a com.android.settings.ACCESSIBILITY_COLOR_SPACE_SETTINGS --user <USER_ID>`
com.android.settings.STORAGE_USB	Forward to primary user.	`adb shell am start -a`

_SETTINGS		`com.android.settings.STORAGE_USB_SETTINGS --user <USER_ID>`
com.android.settings.TTS_SETTINGS	Forward to primary user.	`adb shell am start -a com.android.settings.TTS_SETTINGS --user <USER_ID>`
com.android.settings.USER_DICTIONARY_EDIT	Forward to primary user.	`adb shell am start -a com.android.settings.USER_DICTIONARY_EDIT --user <USER_ID>`
ACTION_CALL	Should invoke handlers within managed profile (if present); otherwise, forward to primary user.	`adb shell am start -a android.intent.action.CALL -d tel:123 --user <USER_ID>`
ACTION_CALL_EMERGENCY	Should invoke handlers within managed profile (if present); otherwise, forward to primary user.	`adb shell am start -a android.intent.action.CALL_EMERGENCY -d tel:123 --user <USER_ID>`
ACTION_CALL_PRIVILEGED	Should invoke handlers within managed profile (if present); otherwise, forward to primary user.	`adb shell am start -a android.intent.action.CALL_PRIVILEGED -d tel:123 --user <USER_ID>`
ACTION_DIAL	Should invoke handlers within managed profile (if present); otherwise, forward to primary user.	`adb shell am start -a android.intent.action.DIAL -d tel:123 --user <USER_ID>`
ACTION_VIEW (Telephony, initiate a phone call)	Should invoke handlers within managed profile (if present); otherwise, forward to primary user.	`adb shell am start -a android.intent.action.VIEW -d tel:123 -c android.intent.category.BROWSABLE --user <USER_ID>`

Cross-profile sharing

The DPC can whitelist sharing of particular content from apps within the personal profile to the managed profile and vice-versa.

Test environment—Device with an inflated work profile.

Test scenarios:

1. Using Test DPC, click **Manage work profile specific policy**.
2. Select **Add cross-profile intents**.
3. Configure the desired flags (as described above and the action, category, and scheme of the intent.
4. Click **Add**.
5. Attempt to share across profiles, depending on the cross-profile intent filter you specified:
    - If it included FLAG_MANAGED_CAN_ACCESS_PARENT, attempt to share the data type specified in your intent filter from a personal app to a work app.
    - If it included FLAG_PARENT_CAN_ACCESS_MANAGED, attempt to share the data type specified in your intent filter from a work app to a personal app.
6. Using Test DPC, click **Manage work profile specific policy** > **Clear all cross-profile intents**.
7. Ensure that any attempt to share across profiles with any data type fails in both directions.

Set app restrictions

App restrictions are a way for an IT administrator to configure an enterprise-specific policy at the individual app level. Examples include enabling or restricting functionality in an app or configuring per-app policies that might apply to specific employees or organizational units. App restrictions can be used to silently configure managed apps using configuration schemas of key-value pairs that app developers have defined in their APK. Learn more about implementing app restrictions.

Test environment—Device with an inflated work profile.

Test scenarios:

1. Using Test DPC, navigate to the **Apps management** section and click **Manage app restrictions**.
2. Select an app from the list that supports app restrictions. (If you need an example of an app with some predefined app restrictions, use Chrome.)
3. Click **Load manifest restrictions** to load any of the restrictions supported by the app. Learn more about Chrome's app restrictions.
4. Click next to the relevant restriction to modify any of the default restrictions:
   - Restrictions with type `Boolean` lets you set it to either `true` or `false`. An example of this from Chrome would be `HomepageIsNewTabPage`.
   - Restrictions with type `Integer` lets you set it to an integer. An example of this from Chrome would be `DefaultCookiesSetting`.
   - Restrictions with type `String` or `String[]` lets you set it to a string or an array of strings. An example of this from Chrome would be `ImagesBlockedForUrls`.
   - Restrictions with type `Bundle` lets you set an array of boolean, integer, string, and string array values. To edit a bundle, first save the name of the key and click **Save**. Click **Edit** to set values in the bundle.
   - Restrictions with type `Bundle[]` lets you set an array of bundles. To edit a bundle, first save the name of the key and click **Save**. Click **Edit** to set values of the bundles within the bundle array.
5. Click **Save** when you're finished modifying the restrictions.
6. Optional. At the top of the screen, click + to add new restrictions.
7. When you're finished adding or modifying the app restrictions, click **Save**.
8. Launch the app to which you passed restrictions.
   - At initial launch, the app retrieves the configuration bundle and modifies its behavior accordingly. It also updates behaviors when the app restrictions are updated.
9. Verify the non-default behavior of the app, based on the restrictions that were applied. From the examples above:
   - If `HomepageIsNewTabPage` is set to `true`, the homepage always displays as a new blank tab. If set to `false`, the homepage is defined by additional app restrictions.
   - If `DefaultCookiesSetting` is, for example, set to `1`, Chrome allows all sites to set local data.
   - If `ImagesBlockedForUrls` is set to `https://www.example.com`, no images display from that URL.

Accessibility services

Changes to accessibility services to support managed profiles allow the DPC to define what services are permitted by setting a whitelist; however, system accessibility services are always available to the user. Accessibility services apply to both profiles; there are no work profile-specific accessibility services.

Test environment—Device with an inflated work profile, and non-system accessibility services installed on the personal side.

Test scenarios:

1. Click **Settings** > **Accessibility**.
2. Try enabling or disabling some accessibility services, and ensure they have the expected result on apps in both the work and personal profiles.
3. Using Test DPC, navigate to the **Accessibility** section and click **Set accessibility services**. (System services should be enabled and greyed out, preventing you from disabling them.)
4. Disable any non-system services and click **OK**.
5. Click **Settings** > **Accessibility** and ensure that non-system services are now disabled and cannot be enabled.

Cross-profile copy and paste

The DPC can enable or prevent a user from pasting content copied from the managed profile into the personal profile. The user can, however, copy content from the personal profile and paste it into the work profile.

Test Environment—Device with an inflated work profile.

Test scenarios:

1. From an app in the work profile**,** copy some text, and ensure you can successfully paste it into an app in the *personal profile*.
2. Using Test DPC, click **Manage work profile specific policy** > **Disallow cross-profile copy/paste** and turn it on.
3. From an app in the work profile, copy some text, and ensure you're *unable* to paste it into an app in the *personal profile*.
4. From an app in the personal profile, copy some text, and ensure you can still successfully paste it into an app in the *work profile*.

Features supported in API level 22 (Android 5.1 Lollipop)

The following features are supported in API level 22 (Android 5.1 Lollipop):

- Show Location Access Status for Managed Profiles in Settings > Location
- Bluetooth headsets in managed profiles
- NFC sharing from the managed profile

Show Location Access Status for Managed Profiles in Settings > Location

This enhancement to the location services screen adds badged location services and a separate item to inform users of their work profile's location status as determined by the policy.

Test environment—Device with an inflated work profile.

Test scenario 1—Location restrictions enabled

1. Using Test DPC, navigate to the **User restriction** section and enable **Disallow share location**.
2. Click **Settings** > **Location**.
3. Toggle **Location services**, and ensure the **Work profile** section remains disabled.
4. Ensure there are no badged items under the **Location services** list.

Test scenario 2—Location restrictions disabled

1. Using Test DPC, navigate to the **User restriction** section and disable **Disallow share location**.
2. Click **Settings** > **Location**.
   a. When the location access switch (at the top of the screen) is off, ensure the **Work profile** section is greyed out and shows **Location** and **Off** in the title and summary.
   b. When the switch is on, ensure the **Work profile** section is not greyed out and that the summary reads On.
3. In both cases, ensure badged items appear in the **Location services** section: Tap on one of the badged items in Location services, and ensure you are taken to a work app rather than a personal app.

Bluetooth headsets in managed profiles

This feature allows Bluetooth headsets to interact with apps running in a managed profile. **Note**: This does not include A2DP.

Test environment—Device with an inflated work profile.

Test scenarios:

1. Pair a Bluetooth headset to your device.
2. Open an app in the managed profile that supports Bluetooth headset usage and attempt to use a feature requiring the headset such as making a call.
3. Unpair and re-pair your headset, and repeat step 2.
4. Disconnect and reconnect your headset, and repeat step 2.
5. Disable and re-enable Bluetooth on your device, and repeat step 2.

NFC sharing from the managed profile

The DPC can now restrict whether NFC can be used to share data from the managed profile to another device.

Test environment—Device with an inflated work profile.

Test scenario 1—NFC beam enabled

1. Using Test DPC, ensure that Disallow outgoing NFC beam is off.
2. Open an app that supports sending data via NFC.
3. Bump that device with another device, ensuring NFC is enabled on both.
4. When prompted, tap to beam data from the app in the managed profile to another device.
5. Ensure the data is successfully transferred.

Test scenario 1—NFC beam disabled

1. Using Test DPC, ensure that **Disallow outgoing NFC beam** is on.
2. Open an app that supports sending data via NFC.
3. Bump that device with another device, ensuring NFC is enabled on both.
4. When prompted, tap to beam data from the app in the managed profile to another device.
5. Ensure that you're unable to transfer the data to the other device.

Features supported in API level 23 (Android 6.0 Marshmallow)

The following features are supported in API level 23 (Android 6.0 Marshmallow):

- Data usage tracking
- Delegated certificate installation
- Work contacts in personal contexts
- Silent enterprise certificate access
- COSU EMM controls
- COSU demo
- Profile owner keyguard management
- Policies for runtime permissions
- Cross-profile app linking

Data usage tracking

A profile or device owner can query for the data usage statistics visible in **Settings** > **Data usage**.

Test environment—Device with an inflated work profile.

Test scenarios:

1. Click **Settings** > **Security** > **Apps with usage access**, and enable **Test DPC**.
2. Within Test DPC, scroll to **Network Data Usage Stats** and perform any of the following, relative steps:
    - Select **Query histories of apps** to view the broken down history of data being used in an app from the managed user from the dates you have specified.
    - Select **Query summary for user** to view the aggregated data usage across all apps for the managed user.
    - Select **Query summary for device** to view the aggregated data usage across all users on the device.

    - Select **Query summary of apps** to see the data usage of each app for the managed user.
3. Compare the returned data from step 2 with what's displayed in **Settings** > **Data usage**, and ensure it's consistent.

Delegated certificate installation

A profile or device owner can now grant a third-party app the ability to call DevicePolicyManager certificate management APIs.

Test environment—Device with an inflated work profile, and an app installed in the profile that has implemented all of the DevicePolicyManager certificate APIs.

Test scenario:

1. Within Test DPC, in the **Certificate management** section, click **Manage delegated certificate installer**, select the correct app from the list, and click **Set**.
2. With the app set as a certificate installer, execute the following **DevicePolicyManager** methods:
    - getInstalledCaCerts
    - hasCaCertInstalled
    - installCaCert
    - installKeyPair
    - uninstallCaCert
    - uninstallAllUserCaCerts
3. Ensure they're executed successfully and don't return security exceptions.

Work contacts in personal contexts

In Android 6.0 Marshmallow, Dialer, and SMS apps in the primary user can now display contacts from the work profile when receiving notifications or viewing past messages or calls. Furthermore, both work and personal contacts are now made available over Bluetooth. These new features can be disabled by policy.

Initiating a call or creating a new message only shows personal contacts, as consistent with the experience in Android 5.0 Lollipop.

Test environment—Device with an inflated work profile (not in device owner mode), dialer, or SMS apps installed in the profile that have implemented the new enterprise Contacts features from Android 6.0 Marshmallow, and contacts in both work and personal system contacts apps. You'll also need a Bluetooth device that syncs contacts.

1. **Test scenario 1**—Cross-profile CallerID enabledUsing Test DPC, click **Manage work profile specific policy** and turn off **Disable cross-profile caller ID**.
2. Dial the number of a corporate contact, and ensure you see the contact picture and name on the in-call screen. **Note:** You cannot search for the corporate contact in the dialer.
3. Receive a call from a corporate contact, and ensure you see the contact picture and name on the in-call screen.
4. Receive a call from a corporate contact when you're using another app, and ensure the notification bar on

top shows the contact name and the picture.

5. Miss a call from a corporate contact, and ensure the "missed call" notification shows the contact name and picture.
6. After a call to or from a corporate contact, ensure the dialer call log displays the number with the contact name and the contact picture.
7. Receive a new SMS from a corporate contact and ensure that the new SMS notification displays the correct corporate name and picture.
8. Send a new SMS to a corporate contact and ensure that corporate contacts show the correct name and picture. **Note**: You cannot search for the corporate contact in the SMS app.
9. Receive a MMS message from the email address of a corporate contact, and ensure the sender is correctly identified and you see the contact picture and name.

Test scenario 2—Bluetooth contact sharing enabled

1. Using Test DPC, click **Manage work profile specific policy** and turn off **Disable bluetooth contact sharing**.
2. Connect a Bluetooth device that syncs contacts from your mobile device.
3. Ensure that both personal and corporate contacts appear in both the contact list and call logs on the Bluetooth device.
4. Place a call to a work contact and receive a call from a work contact on your mobile device. Ensure that in both cases that the correct name appears on the Bluetooth device.
5. Receive an SMS from a corporate contact on your mobile device, and ensure the correct name is displayed on the Bluetooth device.

Test scenario 2—Cross-profile CallerID disabled

1. Using Test DPC, click **Manage work profile specific policy** and turn on **Disable cross-profile caller ID**.
2. Dial the number of a corporate contact, and ensure you don't see the contact picture and name on the in-call screen.
3. Receive a call from a corporate contact, and ensure you don't see the contact picture and name on the in-call screen.
4. Receive a call from a corporate contact when you're using another app, and ensure the notification bar on top does not show the contact name and the picture.
5. Miss a call from a corporate contact, and ensure the "missed call" notification does not show the contact name and picture.
6. After a call to or from a corporate contact, ensure the dialer call-log doesn't show the number with the contact name and the contact picture.
7. Receive a new SMS from a corporate contact and ensure that the new SMS notification doesn't display the correct corporate name and picture.
8. Send a new SMS to a corporate contact and ensure that corporate contacts do not show the correct name and picture. Note: You cannot search for the corporate contact in the SMS app.
9. Receive a MMS message from the email address of a corporate contact, and ensure the sender is not identified and you do not see the contact picture and name.

Test scenario 3—Bluetooth contact sharing disabled

1. Using Test DPC, click **Manage work profile specific policy** and enable **Disable Bluetooth contact sharing**.
2. Connect a Bluetooth device that syncs contacts from your mobile device.
3. Ensure that only personal contacts appear in both the contact list and call logs on the Bluetooth device, and that no corporate contacts appear.

4. Place a call to a work contact and receive a call from a work contact on your mobile device. Ensure that work contact details don't display on the Bluetooth device in either case.
5. Receive an SMS from a corporate contact on your mobile device, and ensure the work contact details are not displayed on the Bluetooth device.

Silent enterprise certificate access

A profile or device owner can now provide a certificate alias silently to a requesting application, thus granting managed apps access to certificates without user interaction.

Test environment—Device with an inflated work profile and an app that requires a client certificate.

Test scenario 1—Silently select certificate

1. Using Test DPC, set your certificate's alias: In the **Certificate management** section, click **Override client certificates with alias**.
2. Use a managed app to prompt for a client certificate, which triggers a callback to Test DPC.
    - The system requests the particular certificate alias required for the action in step 1 from Test DPC.
3. Ensure that the app from step 2 accepts the certificate and behaves accordingly, without prompting the user to select the appropriate certificate.

Test scenario 2—*Do not specify certificate*

1. Using Test DPC, ensure that there is no alias set for **Override client certificates with alias** (under **Certificate management**). If an alias has previously been set, delete the text and click **OK**.
2. Use a managed app to prompt for a client cert, which triggers a callback to the DPC.
    - The system will request the particular certificate alias required for the action in step 1 from Test DPC, but it returns `null`.
3. Ensure that the app from step 2 prompts the user to select the appropriate certificate.

COSU EMM controls

A number of new device-owner capabilities have been introduced to improve the single-use app functionality of Android for Work.

Test environment

Device configured with the device owner, and with two apps installed with the following capabilities:

- App A can enter Lock Task Mode on launch by calling startLockTask. (In Test DPC, apps can also enter this mode by clicking **Lock task** > **Start lock task**.)
- App A can call an explicit intent to launch an activity in App B.
- The activity in App B specified in the explicit intent in App A includes the manifest argument android:lockTaskMode="if_whitelisted" in App B's AndroidManifest.xml

IMPORTANT NOTE: App A should have a way to get out of Lock Task mode by calling stopLockTask. This shouldn't be obvious or documented for end users. If you don't have this backdoor in place, you'll need to factory reset your phone in order to get out of the mode.

Test scenarios:

- With Test DPC set as the device owner, scroll to the **Single-use devices** section and click **Disable keyguard**. Ensure that turning off and on the screen doesn't invoke the keyguard.
- With Test DPC set as the device owner, scroll to the **Single-use devices** section and click **Reenable keyguard**. Ensure that turning off and on the screen invokes the keyguard.
- With Test DPC set as the device owner, scroll to the **Single-use devices** section and click **Disable status bar**. Ensure that you can no longer drag down the status bar, and that no notifications appear.
- With Test DPC set as the device owner, scroll to the **Single-use devices** section and click **Reenable status bar**. Ensure that you can now drag down the status bar, and that notifications appear.
- With Test DPC set as the device owner, scroll to the **User restriction** section and click **Disallow safe boot**. Ensure that when you hold the power button, then long press the Power off dialog, that you are not prompted to reboot into safe mode. **Note**: The instructions to get into Safe Boot are specific to Nexus devices. Your device may use a different method.
- With Test DPC set as the device owner, scroll to the **User restriction** section and disable **Disallow safe boot**. Ensure that when you hold the power button, then long press the Power off dialog, that you are prompted to reboot into safe mode. **Note**: The instructions to get into Safe Boot are specific to Nexus devices. Your device may use a different method.
- With Test DPC set as the device owner, scroll to the **Settings management** section and enable **Keep the device on** while plugged in. Plug in your device, and ensure the screen does not turn off according to any timeout.

Test Scenario: Lock Task mode

1. With Test DPC set as the device owner, scroll to the **Lock task** section, click **Manage lock task list**, and add App A and App B to the lock task list.
2. Launch App A. Ensure that the app launches into Lock Task Mode.
3. Fire the explicit intent from App A to launch into App B, and ensure that App B also launches in Lock Task Mode.
4. Ensure that in Lock Task Mode the user can't leave App A and App B. Ensure the Home button doesn't navigate the user to the default launcher, the Back button doesn't exit either application, and that the app switcher is unavailable.

Profile owner keyguard management

Profile owners can control a subset of keyguard management settings.

Test environment—Device with an inflated work profile.

Test scenario 1—Only show secure notifications

1. In **Settings** > **Sound & notification**, set **When device is locked** to **Show all notification content**.
2. From the keyguard, receive a notification and ensure you can see the entire content of the notification.
3. With Test DPC, turn on **Disable unredacted notifications**.

4. Verify that notifications for the work profile don't display content on the lock screen (Ex: Send an email to the work account and verify that the notification doesn't display with the email content).

Test scenario 2—Disable all notifications

1. In **Settings** > **Sound & notification**, set **When device is locked** to **Show all notification content**.
2. From the keyguard, receive a notification and ensure you can see the entire content of the notification.
3. With Test DPC, turn on **Disable secure notifications**.
4. Verify that notifications for the work profile aren't displayed, at all.

Policies for runtime permissions

Feature Description:

A profile owner or device owner can now manage individual permissions for applications located within the work profile. The DPC can choose to programatically allow, deny, or let the user decide for each permission of a given application. Also, the DPC can choose to auto-accept or auto-deny all future permissions for apps within the work profile.

Test environment—Device with an inflated work profile or in device-owner mode.

Test scenario 1—Setting app-specific permissions

1. With Test DPC, navigate to the **Permission management** section and click **Manage app permissions**.
2. Select an app from the drop-down menu. The screen will be blank if the app doesn't request any permissions, so make sure to choose one that requests at least one permission.
3. Choose an option for each permission:
    a. **Accept** automatically accepts the permission on behalf of the user. The user should not be prompted when the app tries to take the action associated with the permission. For example, if WRITE_CONTACTS app is set to **Accept** for the Contacts app, the user should not be prompted when saving a newly added contact.
    b. **Let user decide** prompts the user to accept or deny the permission when the app tries to take the action associated with the permission. For example, if WRITE_CONTACTS app is set to **Let user decide** for the Contacts app, the user should be prompted when attempting to save a newly added contact.
    c. **Deny** automatically denies the permission on behalf of the user. They should not be prompted when the app tries to take the action associated with the permission. For example, if WRITE_CONTACTS app is set to **Deny** for the Contacts app, the user should be blocked automatically from saving a newly added contact.
4. When you're done adjusting the permissions, click **Back** to save your choices.

Test scenario 2—Setting default permission policy

1. With Test DPC, navigate to the **Permission management** section and click **Get/Set default permission policy**.
2. In the pop-up, choose the default permission option for all apps in the work profile. See above for what those choices mean. **Note**: If you set an app-specific permission after this, the app follows those permission choices.

3. Click **OK** to save your choice.

Cross-profile app linking

Supported links can be opened directly by apps rather than be opened in the browser. For example, maps.google.com clicked in the work profile to be opened by the Google Maps app in the personal profile.

Test environment—Device with an inflated work profile.

Test scenario 1—Policy turned off

1. Using Test DPC, scroll to **User restrictions** > **Set user restrictions** > **Allow web links to apps of the parent** and ensure that it's disabled.
2. Open a supported link using an app in the work profile. The link will open in the work profile in either:
   - The app that supports that link
   - The browser

   If there's no browser or app available, the URL does not open.

Test scenario 2—*Policy turned on*

1. Using Test DPC, scroll to **User restrictions** > **Set user restrictions** > **Allow web links to apps of the parent**.
2. Enable the policy.
3. Using Test DPC, scroll to **Managed profile policy** > **Managed work profile specific policy** > **Add cross-profile intents**.
4. Ensure that there are no cross-profile intents set with the Actions: VIEW and Schemes: http or https.
5. Under **Settings** > **Apps** > ⚙ > **App links** > (any app), you see the **Open supported links** option. When you click this option, it displays a drop-down list with three choices (nicknames for ease of explaining): Open in this app (always), Ask every time (ask), and Don't open in this app (never).
6. Open a supported link using an app in the work profile.
7. Follow the table below to see what the expected behavior should be:

Apps installed in the personal profile and their supported links option	Apps installed in the work profile and their supported links option	Expected behavior
One app: ask	No apps available	Resolve in the personal profile
One app: always	No apps available	Resolve in the personal profile
One app: never	No apps available	The intent cannot be resolved
One app: ask	One app: ask	Ask the user which app they would like to use in which profile
One app: always	One app: ask	Resolve in the personal profile
One app: never	One app: ask	Resolve in the work profile

One app: ask	One app: never	Resolve in the personal profile
One app: always	One app: never	Resolve in the personal profile
One app: never	One app: never	Resolve in the personal profile
One app: ask	One app: always	Resolve in the work profile
One app: always	One app: always	Resolve in the work profile
One app: never	One app: always	Resolve in the work profile
Two apps: always	No apps available	Resolve in the personal profile with the latest app changed to always
A browser	No apps available	The intent cannot be resolved
A browser	A browser	Resolve in the work profile
Browser and One app: ask	No apps available	Ask the user which app they would like to use in the personal profile
Browser and One app: always and One app: ask	No apps available	Resolve in the personal profile with the app changed to always

Document Changelog

Version	**Date**	**Description**
1.0	October 22, 2015	Initial Release