Disabling GMS features using MX

Devices with Google Mobility Services (GMS) are becoming increasingly popular in the Enterprise space, these are devices which have Google's proprietary suite of applications and services built in and can provide your enterprise deployment with a powerful foundation on which to build your line of business applications and workflow. Exactly why an enterprise would choose GMS over non-GMS (also called the Android Open Source Project, AOSP) is outside the scope of this blog: you might wish to deploy "managed" Android devices (https://enterprise.google.com/android/solutions/purpose-built/), you may be swayed by the additional protection offered by Play Protect (https://www.android.com/play-protect/) or your applications might simply require Google Maps and enhanced location precision but whatever your reasons, the prerequisite assumption is you have chosen a deployment incorporating GMS devices.

Although GMS incorporates an ever expanding range of features, some of the most popular features are listed below:

- Google Play Services APIs including:
    - Firebase Cloud Messaging
    - Fused Location API
    - Geocoding API
    - Google Maps API
    - Google Cast API
- Google Play Store
- Google Play Protect
- Google Chrome (& safe browsing)
- Gmail
- Voice recognition

Having chosen GMS devices, many customers find they wish to selectively block certain functionality and applications; this blog aims to inform the administrator exactly how functionality can be selectively disabled and the implications of doing so.

Why disable GMS features at all?

GMS features as an overall suite are extremely powerful, so much so that arguably a consumer smartphone is not fit for purpose without them. As we see Google increasingly concentrate on the Enterprise market, more and more enterprise functionality will make its way into play services for example the managed play store or support for managing purpose-built devices.

You may wish to selectively disable GMS features to:

- Prevent automatic application updates from the Play Store
- Prevent the user from accessing specific functionality e.g. the News app
- Control whether your devices location is visible to Google by e.g. disabling the fused location API
- Prevent some 'Share' options being displayed to the user e.g. prevent sharing to drive to avoid the user copying off sensitive data
- Prevent access to Google+ as it is not relevant to the employee's work.

And many other use cases

How to disable GMS features

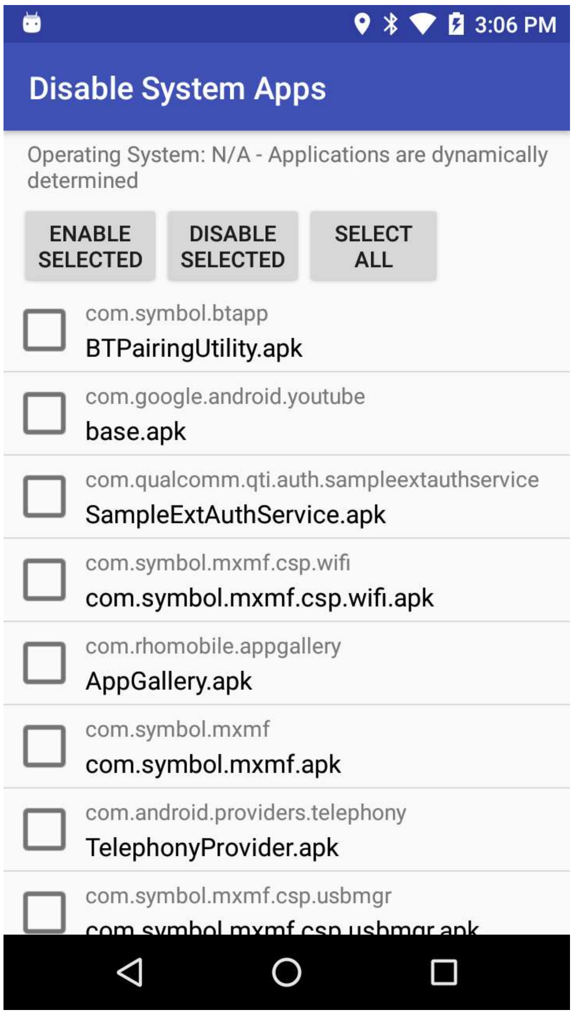
There are 3 steps to disabling GMS features:

1. Determine which packages represent the features you wish to disable
2. Disable these packages using the MX App Manager (http://techdocs.zebra.com/emdk-for-android/6-3/mx/appmgr/); as with any Zebra 'manager' this can be called by stage now, during device configuration or at runtime through our EMDK API.
3. Restart the device

Determining which packages to disable

Unfortunately there is not a public reference on which Android packages are included in GMS devices and the functionality of each, there will therefore unfortunately be some degree of trial and error to determine which packages require disabling. To add to this, the packages which comprise a GMS build vary from desert to desert so you may need separate configurations for each desert if your deployment contains different android versions.

To aid in the 'trial and error' I have put together an application which lists all the system apps installed on your Zebra device and allows you to disable or enable them: https://github.com/darryncampbell/Disable-System-Apps , there is also the ability to define your own set of applications / package names and for more information see the readme (https://github.com/darryncampbell/Disable-System-Apps/blob/master/README.md):


3:06 PM

Disable System Apps

Operating System: N/A - Applications are dynamically determined

com.symbol.btapp
BTPairingUtility.apk

com.google.android.youtube
base.apk

com.qualcomm.qti.auth.sampleextauthservice
SampleExtAuthService.apk

com.symbol.mxmf.csp.wifi
com.symbol.mxmf.csp.wifi.apk

com.rhomobile.appgallery
AppGallery.apk

com.symbol.mxmf
com.symbol.mxmf.apk

com.android.providers.telephony
TelephonyProvider.apk

com.symbol.mxmf.csp.usbmgr
com.symbol.mxmf.csp.usbmgr.apk

To determine which packages are installed on your device then you can run the following command with your device connected to adb:

```
adb shell "pm list packages -f"
```

You can of course combine this with grep for a more concise output:

```
Adb shell "pm list packages -f | grep com.android
```

```
You will get an output something like this (obtained from my TC51 GMS):
package:/system/priv-app/BTPairingUtility/BTPairingUtility.apk=com.symbol.btapp
package:/data/app/com.google.android.youtube-1/base.apk=com.google.android.youtube
package:/system/app/SampleExtAuthService/SampleExtAuthService.apk=com.qualcomm.qti.auth.sample
extauthservice
package:/system/priv-
app/com.symbol.mxmf.csp.wifi/com.symbol.mxmf.csp.wifi.apk=com.symbol.mxmf.csp.wifi
package:/system/priv-app/AppGallery/AppGallery.apk=com.rhomobile.appgallery
package:/system/priv-app/com.symbol.mxmf/com.symbol.mxmf.apk=com.symbol.mxmf
package:/system/priv-
app/TelephonyProvider/TelephonyProvider.apk=com.android.providers.telephony
package:/system/priv-
app/com.symbol.mxmf.csp.usbmgr/com.symbol.mxmf.csp.usbmgr.apk=com.symbol.mxmf.csp.usbmgr
package:/data/app/com.afwsamples.testdpc-1/base.apk=com.afwsamples.testdpc
package:/data/app/com.google.android.googlequicksearchbox-
1/base.apk=com.google.android.googlequicksearchbox
package:/system/priv-app/CalendarProvider/CalendarProvider.apk=com.android.providers.calendar
package:/data/app/com.zebra.datawedgeexerciser-1/base.apk=com.zebra.datawedgeexerciser
package:/system/priv-
app/com.symbol.mxmf.csp.xmlmgr/com.symbol.mxmf.csp.xmlmgr.apk=com.symbol.mxmf.csp.xmlmgr
package:/system/priv-app/MediaProvider/MediaProvider.apk=com.android.providers.media
package:/system/priv-
app/GoogleOneTimeInitializer/GoogleOneTimeInitializer.apk=com.google.android.onetimeinitialize
r
package:/system/app/ModemTestMode/ModemTestMode.apk=com.qualcomm.qti.modemtestmode
package:/system/priv-
app/com.symbol.mxmf.csp.wirelessmgr/com.symbol.mxmf.csp.wirelessmgr.apk=com.symbol.mxmf.csp.wi
relessmgr
package:/system/app/shutdownlistener/shutdownlistener.apk=com.qualcomm.shutdownlistner
package:/system/priv-app/WallpaperCropper/WallpaperCropper.apk=com.android.wallpapercropper
package:/system/priv-app/CNEService/CNEService.apk=com.quicinc.cne.CNEService
```

I will attach my full output from my TC51 to the bottom of this blog. Many of these will be self-explanatory, for example:
Com.google.android.apps.plus disables the Google+ application and the ability to Share to Google+
Com.android.location.fused disables the fused location provider available through the Location Services API

And some may be less obvious, for example:
Com.google.android.googlequicksearchbox disables the ability to web search for a highlighted word outside of Chrome

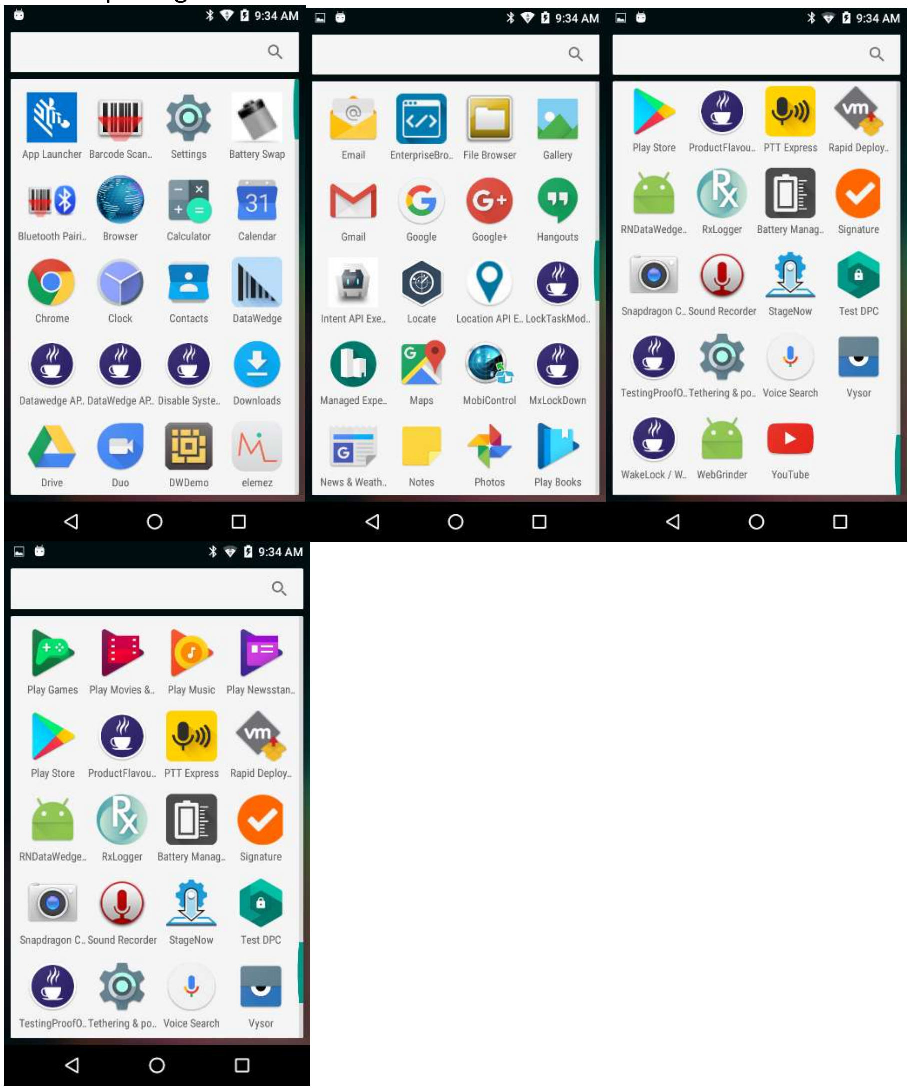
Disable these packages using the MX App Manager

The MX app manager (http://techdocs.zebra.com/emdk-for-android/6-3/mx/appmgr/) can be used to disable specific packages using the following profile:

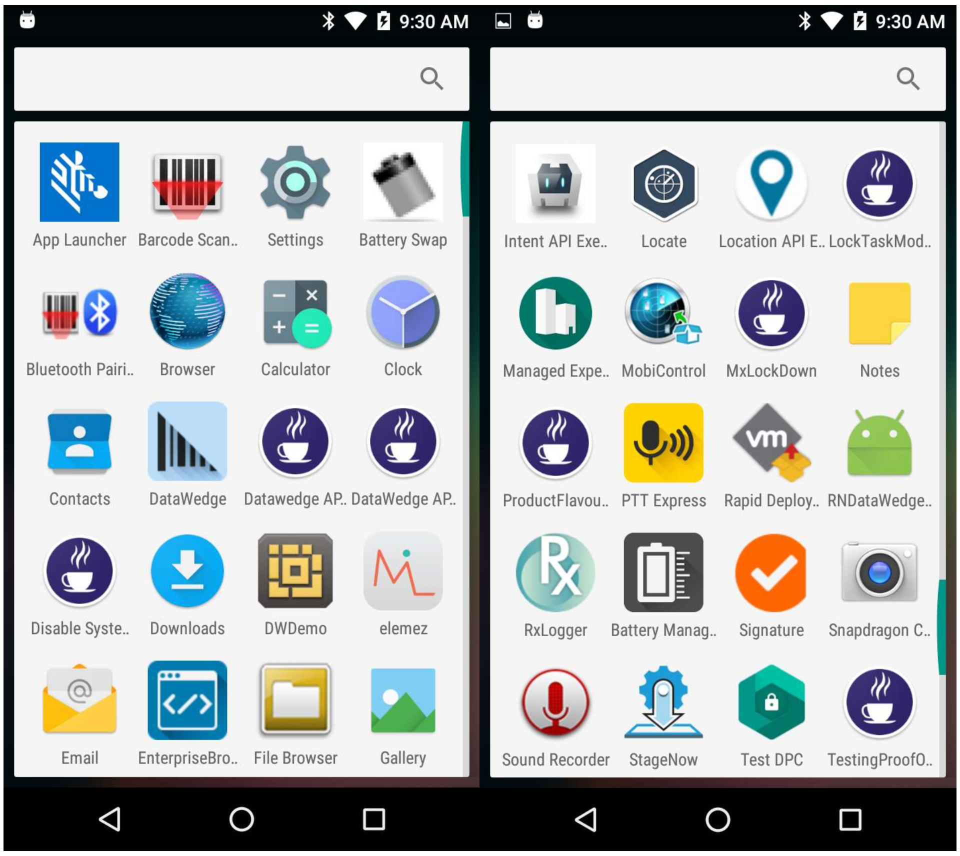
```
<characteristic type="Profile">
  <parm name="ProfileName" value="ApplicationManager"/>
  <parm name="ModifiedDate" value="2017-01-25 08:07:34"/>
  <parm name="TargetSystemVersion" value="6.0"/>
  <characteristic type="AppMgr" version="4.4">
    <parm name="emdk_name" value="AppManager"/>
    <parm name="Action" value="DisableApplication"/>
    <parm name="Package" value="com.android.sampleApplication"/>
  </characteristic>
</characteristic>
```

As mentioned earlier, you can initiate this profile as part of your device configuration (via StageNow) or at runtime using the EMDK. If you are using the test application at https://github.com/darryncampbell/Disable-System-Apps then this functionality is part of that application.

Here is a before and after of disabling all GMS applications on my TC51 Marshmallow device:
All GMS packages enabled:



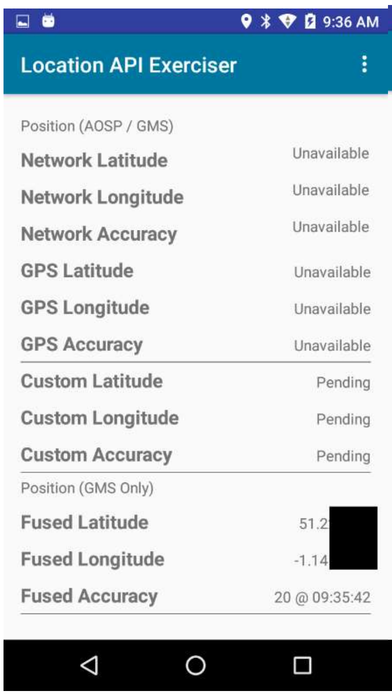
All GMS packages disabled:



There are 6 different types of applications installed on my device:

- Applications which are part of GMS, these have disappeared between and include Chrome, Maps, Gmail etc.
- Applications which are part of the standard Android AOSP continue to be present, e.g. File Browser, Gallery and Settings
- Applications which are pre-loaded by Zebra as part of the Zebra BSP and provide value adds to Zebra users e.g. the Bluetooth pairing utility, StageNow. These remain unaffected.
- Applications I have developed myself and side-loaded to the device, with the common coffee cup icon remain unaffected.
- Applications downloaded from the Google Play store, e.g. ZXing (Barcode Scanner): These remain on the device despite the play store itself being disabled but are no longer subject to being automatically updated.
- Applications available from Zebra which have been post-loaded onto the device, e.g. Enterprise Browser remain unaffected.

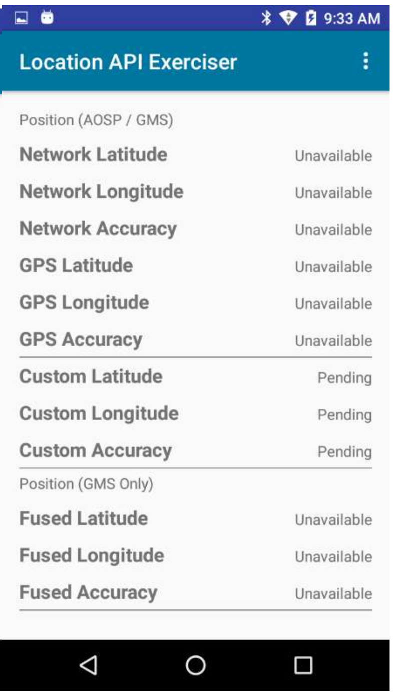
Applications which rely on GMS services such as the fused location provider no longer report a fused position, as evidenced from this before / after screenshot of my location API exerciser, https://github.com/darryncampbell/Location-API-Exerciser.



9:36 AM

Location API Exerciser

Position (AOSP / GMS)	
Network Latitude	Unavailable
Network Longitude	Unavailable
Network Accuracy	Unavailable
GPS Latitude	Unavailable
GPS Longitude	Unavailable
GPS Accuracy	Unavailable
Custom Latitude	Pending
Custom Longitude	Pending
Custom Accuracy	Pending
Position (GMS Only)	
Fused Latitude	51.2[redacted]
Fused Longitude	-1.14[redacted]
Fused Accuracy	20 @ 09:35:42



9:33 AM

Location API Exerciser

Position (AOSP / GMS)	
Network Latitude	Unavailable
Network Longitude	Unavailable
Network Accuracy	Unavailable
GPS Latitude	Unavailable
GPS Longitude	Unavailable
GPS Accuracy	Unavailable
Custom Latitude	Pending
Custom Longitude	Pending
Custom Accuracy	Pending
Position (GMS Only)	
Fused Latitude	Unavailable
Fused Longitude	Unavailable
Fused Accuracy	Unavailable

Note: I have redacted some of the location precision for privacy.

Restart the device

Notice how Google Duo is available in the screenshot illustrating all the GMS packages being enabled, this is curious behaviour since Duo does not appear after a device restart on TC51. Similarly, I have observed other strange behaviour after enabling some packages e.g. the photos application failed to fully start. For these reasons, I advise rebooting your device after making any changes to the enabled or disabled packages.

Other considerations

- Enterprise Browser on Marshmallow will continue to work even after "com.google.android.webview" has been disabled. I am surprised by this since EB depends on the system webview but this behaviour may change as the OS is updated to Nougat and 'O' where the webview provider is provided by the Chrome APK.
- As far as I know the list of applications which are part of GMS for each OS flavour is proprietary information to Google so whilst it can be fairly straight forward to determine through trial and error, I cannot go into extensive detail in this blog. If you require the list of GMS applications on each OS flavour, work with your sales engineer who may be able to help.
- Android's Device Policy Manager API (https://developer.android.com/reference/android/app/admin/DevicePolicyManager.html) contains several methods related to enabling system apps. This is complementary but separate from the enabling and disabling system apps discussed in this blog and is concerned with system apps disabled whilst provisioning an Android managed device.