Migration, Deployment, and General Supplement

Workspace ONE for Android

Android applications can be integrated with the Workspace ONE® platform, by using its mobile software development kit.

This document is a collection of content from the earlier developer guide that hasn't yet been moved to the new integration guide set.

Table of Contents

SQLCipher Support and Migrating Databases....2

Required Libraries and Dependencies for Migration to the Latest SDK Version....3

Application Signing Key Registration....6

SDK Profile Refresh Interval....14

Run a Process Before Initialization, Optional....15

Certificate Pinning....16

Base Activity....17

Push Notification Support....20

APIs to Use Custom Certificates for Your SDK-Built Apps....27

Set Inactivity Timer Reset for Application Inactivity....29

Document Information....30

SQLCipher Support and Migrating Databases

The Workspace ONE SDK for Android uses SQLCipher 4.4 by default. In an application upgrade scenario, the SDK handles database migration. However, if an application uses additional databases, you must migrate them.

To migrate and upgrade an existing database and preserve data and schemas, use the new default settings and use PRAGMA cipher_migrate. For more information, access SQLCipher 4.0.0 Release.

Required Libraries and Dependencies for Migration to the Latest SDK Version

Add the necessary libraries and add dependencies to Gradle when you migrate to the latest Workspace ONE SDK for Android. Also, ensure that the base classes have the latest code. Always upgrade the dependent libraries with the SDK because the dependent libraries that are packaged with the SDK are upgraded with every new version of the SDK.

20.2

Workspace ONE SDK for Android has migrated to Koin version 2.1.0. This version is not compatible with earlier Koin versions. Any app consuming Workspace ONE SDK for Android 20.3 must update to Koin v2.1.0 or later.

19.8

Workspace ONE SDK for Android offers a new method getNightMode() to support Dark Mode, also known as Night Mode. Dark Mode includes colors and style updates in the AWFramework. After migration, validate that your app's colors and styles are not having UI issues. If you need specific colors from AWFramework, and the migration to 19.9 (or later) changes those colors, you can revert the colors back. Refer to the 19.8 colors.xml file and override those colors in the app's colors.xml file. You can also revert styles back in the same way.

19.7

Workspace ONE SDK for Android v19.8 uses AndroidX support libraries. AndroidX replaces the original support library APIs with packages in the AndroidX namespace. Refer to Migrating to AndroidX for steps to migrate to AndroidX. You must migrate to AndroidX to use Workspace ONE SDK for Android v19.8.

19.4

The Workspace ONE SDK for Android v19.6 adds support for tunnelling features on Android 10 (Q). Older SDK versions fail with a "Connection reset" or "Tunnel connection failed" error on Android 10 (Q) devices. Applications using tunnelling features should use SDK versions 19.6 or later and use the AW wrapper classes provided by the AWNetworkLibrary. For more details, see the separate Workspace ONE for Android Networking Developer Guide.

17.x

The Workspace ONE SDK for Android does not require entries to migrate from 17.x.

16.10

Add the following entry to the build.gradle file.

```
android {
   defaultConfig {
   …  ndk {
            abiFilters ‘x86’, ‘x86_64’, ‘armeabiv7a’, ‘arm64-v8a’
            }
         }
      }
```

The Workspace ONE SDK for Android 17.x includes updates to the AWNetworkLibrary. The updates remove the requirement to send an authentication token in an HTTP header. See the separate Workspace ONE for Android Networking Developer Guide for updated requirements.

16.x

Select a migration process based on the use of a login module for initialization.

- Login Module - To upgrade the master key manager, override the getPassword method in the application class. This override extends AWApplication to handle the upgrade.

```
    @Override
    public byte[] getPassword() {
        if
    (SDKKeyManager.getSdkMasterKeyVersion(context
    )
            !=
    SDKKeyManager.SDK_MASTER_KEY_CURRENT_VERSION)
    {

            SDKKeyManager.newInstance(context);
        }
        return super.getPassword();
    }
```

- No Login Module - Initialize your SDKContextManager and call the updateSDKForUpgrade() API.

```
    SDKContextHelper sdkContextHelper = new SDKContextHelper();
    SharedPreferences securePreferences =
    SDKContextManager.getSDKContext().getSDKSecur ePreferences();
    int oldSDKVersion = securePreferences.getInt(SDKSecurePreferencesKeys.CURRENT_FRAMEWORK_VERSION, 0);
    SDKContextHelper.AWContextCallBack callBack = new SDKContextHelper.AWContextCallBack() {
        @Override
        public void onSuccess(int requestCode, Object result) {
            //success continue
        }

        @Override
        public void onFailed(AirWatchSDKException e) {
            //failed
        }
    };
    try {
        sdkContextHelper.updateSDKForUpgrade(0, oldSDKVersion, callBack);
    } catch (AirWatchSDKException e){
            //handle exception
    }
```

Older than 16.x

- Libraries
  - A total of 23 libraries including JAR and AAR files
  - SQLCipher library is an AAR file instead of a JAR file
- Gradle Methods
  - For 16.02 – compile (name:‘AWFramework 16.02’,ext:‘aar’)
  - For 16.04 – compile (name:‘AWFramework 16.04’,ext:‘aar’)
  - For both – compile (name:‘sqlcipher-3.5.2–2’,ext:‘aar’)
- Code

  If you are not using the login module for initialization, check the implementation of base classes.

Application Signing Key Registration

In some cases, the public signing key used to sign an app must be:

- Registered with VMware.
- Configured in the customer Workspace ONE UEM management console.

Registration Requirement

The need for key registration might apply to an application that has integrated the Workspace ONE software development kit (SDK) for Android, depending on:

- The version of SDK that is integrated.
- How the app will be installed.
- Whether the app will be installed into an Android Enterprise partition.

An enterprise that sets up an Android Enterprise partition on a device will control what apps can be installed into the partition. The mechanism of control can be a managed Play Store, or a private enterprise Play Store, for example. In any case, the requirement for signing key registration is the same.

As follows.

- If the app will be installed into an Android Enterprise partition, and the SDK is version 19.4 or earlier, then the key must be registered.
- If the app will be installed into an Android Enterprise partition, and the SDK is version 19.6 or later, then the key doesn't have to be registered.
- If the app will be uploaded to the customer Workspace ONE UEM management console, and installed by the Workspace ONE Intelligent Hub app, then the key doesn't have to be registered. This applies regardless of the SDK version.

  In this case some security restrictions must be relaxed on the UEM in order to allow installation of an app that is under development. Specifically, the following must be enabled in the Restrictions payload of the assigned device profile:

  - Allow USB Debugging.
  - Allow NonMarket.
- If the app will be installed from the public Play Store, without enterprise control, then the key must be registered. Note:

  - This applies regardless of the SDK version.
  - Installation of this type won't be into an Android Enterprise partition.
  - This will include any device enrolled in Workspace ONE registered mode.
  - This will include any device enrolled using Workspace ONE legacy Android enrollment.

For details of the managed Play Store, see the Android developer website, for example:
https://developer.android.com/distribute/google-play/work

Registration Overview

If the key must be registered, then follow the Registration Procedure.
Start the registration process as soon as possible, in parallel with your development effort. Finalization of registration at VMware will depend on releases of the Workspace ONE Intelligent Hub app and the Workspace ONE SDK for Android. Expect the process to take from two to six weeks.

After registration at VMware, some configuration steps must be taken in the customer UEM. The UEM configuration steps are included in the Registration Procedure, towards the end.

The configuration steps would also apply to an app from a public independent software vendor (ISV) or other partner. A list of partner app signing keys is included here, see ISV App Keys.

Registration Procedure

Proceed as follows.

1. Send the public signing key to your Workspace ONE UEM support representative for registration at VMware. This action adds the key to releases of the Workspace ONE Intelligent Hub and the Workspace ONE SDK for Android. This is the step that takes two to six weeks.

   You can contact your representative by creating a support request on the Workspace ONE support website:
   https://support.workspaceone.com/

2. Extract the signing key value from the APK.
    1. Start a terminal or other command-line application on a machine that has the app APK file.
    2. Navigate to the directory containing the APK file.
    3. Display the signing key value, for example by running keytool as follows.

        ```
        keytool -rfc -printcert -jarfile YourAppName.apk
        ```

    4. Copy and paste the key signing value into a text editor.
    5. In the text editor, remove all line breaks and spaces so the value is one, long string value.

    The required value is all the text between the begin and end markers, as in:

    —--BEGIN CERTIFICATE---AllTextHere—--END CERTIFICATE---

    This text will be the signing key value utilized in the next step.

3. Add the signing key value to the SDK profile assigned to the Workspace ONE Intelligent Hub for Android. This requires administrator privileges. Proceed as follows.

    1. Find the SDK profile assigned to the Workspace ONE Intelligent Hub by navigating as follows in the Workspace ONE UEM console:
    Groups & Settings, All Settings, Devices & Users, Android, Intelligent Hub Settings, SDK Profile.

        Make a note of the name of the assigned profile. The name will be required in the next step.

    2. Navigate to the named Custom Settings payload of the SDK profile that is assigned to Workspace ONE Intelligent Hub. It is either the default profile or a custom profile.

        - Default SDK profile would be at:
        Groups & Settings, All Settings, Apps, Settings and Policies, Settings, Custom Settings.
        - Custom SDK profile would be at
        Groups & Settings, All Settings, Apps, Settings and Policies, Profiles.

        Then select the profile and the Custom Settings payload.

    3. To introduce the trusted key signing value in the Custom Settings text box, enter a code block like the following.

        ```
        {
            "trustedKeys" :["key_value_1","key_value_2"]
        }
        ```

    4. Replace, for example, “key_value_1” with the key signing value. To add multiple trusted keys, separate the values with a comma.

    5. Save the profile.

4. Add the signing key value to the SDK profile assigned to your application. This requires administrator privileges. Proceed as follows.
    1. Find the SDK profile assigned to your application by navigating as follows in the Workspace ONE UEM console:
    Apps & Books, Native, Public, then select the application.

    Then continue navigation to: More, SDK, SDK Profile.

    Make a note of the name of the assigned profile. The name will be required in the next step.
    2. Navigate to the named Custom Settings payload of the SDK profile that is assigned to your app. It is either the default profile or a custom profile.
        - Default SDK profile would be at:
        Groups & Settings, All Settings, Apps, Settings and Policies, Settings, Custom Settings.
        - Custom SDK profile would be at
        Groups & Settings, All Settings, Apps, Settings and Policies, Profiles.

        Then select the profile and the Custom Settings payload.
    3. To introduce the trusted key signing value in the Custom Settings text box, enter a code block like the following.

        ```
        {
            "trustedKeys" :["key_value_1","key_value_2"]
        }
        ```
    4. Replace, for example, “key_value_1” with the key signing value. To add multiple trusted keys, separate the values with a comma.
    5. Save the profile.
5. Update your app with the latest version of the SDK that includes the registered signing key.
6. Update the device to the latest version of Workspace ONE Intelligent Hub that includes the registered signing key.
7. Test the public application on a device enrolled with the Workspace ONE Intelligent Hub.

This concludes the registration procedure.

ISV App Keys

Cut and paste these app keys to the SDK profiles of the Workspace ONE Intelligent Hub and the app if you use apps that are from public independent software vendors (ISV) or partner apps.

Note: Each key must be pasted without line breaks. It can be helpful first to copy the key from here into a text editor. Then delete any line breaks in the editor before copying again and pasting into the management console user interface.

Webalo

MIIDdDCCAlygAwIBAgIETczrszANBgkqhkiG9w0BAQUFADB8MQswCQYDVQQGEwJVUzETMBEGA1UECBMKQ2FsaWZvcm5pYTEUMBIGA1U
EBxMLTG9zIEFuZ2VsZXMxFTATBgNVBAoTDFdlYmFsbywgSW5jLjEUMBIGA1UECxMLRGV2ZWxvcG1lbnQxFTATBgNVBAMTDFdlYmFsby
wgSW5jLjAeFw0xMTA1MTMwODI4MzVaFw0zODA5MjgwODI4MzVaMHwxCzAJBgNVBAYTAlVTMRMwEQYDVQQIEwpDYWxpZm9ybmlhMRQwE
gYDVQQHEwtMb3MgQW5nZWxlczEVMBMGA1UEChMMV2ViYWxvLCBJbmMuMRQwEgYDVQQLEwtEZXZlbG9wbWVudDEVMBMGA1UEAxMMV2Vi
YWxvLCBJbmMuMIIBIjANBgkqhkiG9w0BAQEFAAOCAQ8AMIIBCgKCAQEAonjegLiAInGofaVq94A+lVA0OkDNjpNSTirUHw0sM4EMvyO
KvgUGruYI1qKP0URiOqkxyeywMWuGXrQTYygiIaLgJFA3Az9Pgx/oWFkYaxNMzbXZpy05EFgRRxJoEt7c5PA7Bk+syvOqm0cuob39wk
M0wDK5opjqIhB6kSdwDCt4F6omjenTTzydlWNpJxCjEqUE78F3Cydqkh8VAciIGXaco/T9QXDZxHm4GAt0XicGBtnBEi1lGSWn0+le8
NHy6e/H/uxKZe3faiEDO8uwGxdYQRw+4hYhUsdFB+DN1XgO9RX7xgnUYuSFIXS6/950EqLVm8LBFt+vMZjOIfd7awIDAQABMA0GCSqG
SIb3DQEBBQUAA4IBAQBosVL3vCn+dHqqwHSEnsOMuHjeacgB2nNNjRZ78fpL795BbM8DjLnUh1ialL32AC6t/VaVldnKoO6iFNSk+UO
xn6I9wHn1ZGwLrSwO+G+uOCp1H23MVk3FuIzHeYRQc5csgFUkMsFgFWd9lkf0sRXLai8NXEYYhZTxdG5BNixx2EbB5QqVPDaODomvT6
WsCyeWzTKzd3RcuOa9nnbJ5B5343QoEq78UxYZL+BbdaYWYx9jU5z/PUt/ZxCQx4yBrhnBwYPUiUGtLL4zgvYEpHdK3qa7O8mJNkunF
Wx4oNycFLPdzWo1q2nYL1FOCo0eXJVGUUgpShtaYieBuf6Tyydi

PrinterOn

MIICbzCCAdigAwIBAgIETYjhPzANBgkqhkiG9w0BAQUFADB8MQswCQYDVQQGEwJDQTEQMA4GA1UECBMHT250YXJpbzESMBAGA1UEBxM
JS2l0Y2hlbmVyMRYwFAYDVQQKEw1QcmludGVyT24gSW5jMRQwEgYDVQQLEwtEZXZlbG9wbWVudDEZMBcGA1UEAxMQQW5ndXMgQ3Vubm
luZ2hhbTAeFw0xMTAzMjIxNzQ5NTFaFw0zODA4MDcxNzQ5NTFaMHwxCzAJBgNVBAYTAkNBMRAwDgYDVQQIEwdPbnRhcmlvMRIwEAYDV
QQHEwlLaXRjaGVuZXIxFjAUBgNVBAoTDVByaW50ZXJPbiBJbmMxFDASBgNVBAsTC0RldmVsb3BtZW50MRkwFwYDVQQDExBBbmd1cyBD
dW5uaW5naGFtMIGfMA0GCSqGSIb3DQEBAQUAA4GNADCBiQKBgQCRJ+Vt29LUH3x2A4lIS5xPBYBpsZculN9imd5fo/ambJigE/a4KEh
ULpH7rvpYd41GgOfyQTFsVdszAjn5t8bWANvaFmZBNaYByRuVzq3XyflCodHQwLmF9rV3l44ug+e5Z9BOSJiAEtFdoAWiQjI5HzxDo+
kfUtexw5QmU14cSwIDAQABMA0GCSqGSIb3DQEBBQUAA4GBAChFPQFlwmqiVBGECvRA79XnVZjlPGRc33FtkSYBQwNDj25NB/yisuFk0
1ArI8UNr5IqqidkhTjQq9EQatMveXxc5ro3SM7kWj7VorXP9fbgGEt1L+3J0Me8ca7GjPSB1lzOq2si9dfoK3N+g+bL7OP1yeq6jEHm
ruHU2VhbZSuf

Capriza / Worksimple

MIIDbzCCAlegAwIBAgIEB65LmjANBgkqhkiG9w0BAQsFADBoMQswCQYDVQQGEwJJTDEPMA0GA1UECBMGSXNyYWVsMRUwEwYDVQQHEwx
Ib2QgSGFzaGFyb24xEDAOBgNVBAoTB0NhcHJpemExDDAKBgNVBAsTA1JuRDERMA8GA1UEAxMIRG9yIFR6dXIwHhcNMTQwNTE5MDc0ND
I4WhcNNDExMDA0MDc0NDI4WjBoMQswCQYDVQQGEwJJTDEPMA0GA1UECBMGSXNyYWVsMRUwEwYDVQQHEwxIb2QgSGFzaGFyb24xEDAOB
gNVBAoTB0NhcHJpemExDDAKBgNVBAsTA1JuRDERMA8GA1UEAxMIRG9yIFR6dXIwggEiMA0GCSqGSIb3DQEBAQUAA4IBDwAwggEKAoIB
AQCF0AyIYm25y63pP5257bvWsm9QMt97LF4/CdACq9Btt39uyagI58dbc48WY4WCpQFX0tkxWkkxBlWNVSPkuAyxzAq46BJuYIT13oS
EUJbvYVFtvoh9iJM9wldczPLbsMBO0KQ2iouJNU3qAIQGMBAoKW1IK9SHxW9XkHxXoew7sb9eLhnkMycHjLaJNs6HsqdXrDYQRFZCqO
Fdh/FI1bHHtUVzLcKsvLN09Z1nNvLe3WtiKvR7dcPmonv9khgjOTN9hNp3QGzUla87hfxNwzrvoyH3zdRBIEI4Y7jVC1foMepr7wDB3
EHwkv+qx88aRSw7zxRoo86LaLybj5wrVCAfAgMBAAGjITAfMB0GA1UdDgQWBBSd7+aB2HEOxA30rE/qPU8FXTIWOzANBgkqhkiG9w0B
AQsFAAOCAQEAOdnOSEXj3X0Fyypaelzc0ShBBtAZUPZkYS+FUblLxG058FG5Z3lhz1HENJFMY6GaBx9bzGJmlf8GB/IdrAGbK6aC7KT
xa4CVa//QUtWkEN+8tTsbU2FjNXwhrkwCLGSYBHk5HeePPKG+ZLjib93mFuM2/q8AE2gpgijHKgag08Uvg6edR4w3wCUnFz4E+lRFo2
lp9vmpsjkTkUoc+eessX2+mUJLEZqyztybtJXh52qyUxSjEvbmralLf7a5LRk20tyrMclimxBxkG3L91uTCJn4bILkklYX5cWLkxLwd
e7bacISiUjWrzVsYHxPE/6QuDESu1r7G+I9k+xyAZUTSg==

Varonis / DatAnywhere

MIICBTCCAW6gAwIBAgIEUVwUNDANBgkqhkiG9w0BAQUFADBHMQ8wDQYDVQQGEwZJc3JhZWwxEDAOBgNVBAoTB1Zhcm9uaXMxEDAOBgN
VBAsTB1Zhcm9uaXMxEDAOBgNVBAMTB1Zhcm9uaXMwHhcNMTMwNDAzMTEzNjIwWhcNNDMwMzI3MTEzNjIwWjBHMQ8wDQYDVQQGEwZJc3
JhZWwxEDAOBgNVBAoTB1Zhcm9uaXMxEDAOBgNVBAsTB1Zhcm9uaXMxEDAOBgNVBAMTB1Zhcm9uaXMwgZ8wDQYJKoZIhvcNAQEBBQADg
Y0AMIGJAoGBAKG8QUrZAH8JtndsFArybDkvNjfT6HrMrytzZQGtXWkk063tIQZZm1g24UVGblbwReschTEStops6eD9WAk/gnosR+rj
XiDGP6M6MyDJJRke08BuRgDFFdGZDrOVkAzBcf3ikkHVUw+6JpjtNpcC70DUyNyVeyvd4fuAlKXttYbVAgMBAAEwDQYJKoZIhvcNAQE
FBQADgYEANP3fmpRobzSONlxKB7ckmyK7PUbjmXh97Fanwtqfiw/a2MDAhkoDEG3Z7BPxfPWdjRQIxcxo3UaAUolXhszYKEtIS7t9z/
G1tIlfIZtDWHgElZc243uvvHr/lMXU5OQBUPNwjOSeJivbFnYUDe0Omk+1NNFTCZQJuT9ses1n/Qs=

Palo Alto Networks / Traps

MIIDnTCCAoWgAwIBAgIEWn+4xjANBgkqhkiG9w0BAQsFADB/MQswCQYDVQQGEwJVUzELMAkGA1UECBMCQ0ExFDASBgNVBAcTC1NhbnR
hIENsYXJhMSAwHgYDVQQKExdQYWxvIEFsdG8gTmV0d29ya3MsSW5jLjEOMAwGA1UECxMFVHJhcHMxGzAZBgNVBAMTElBhbG8gQWx0by
BOZXR3b3JrczAeFw0xNjA3MDYxMjQzNDhaFw00MzExMjIxMjQzNDhaMH8xCzAJBgNVBAYTAlVTMQswCQYDVQQIEwJDQTEUMBIGA1UEB
xMLU2FudGEgQ2xhcmExIDAeBgNVBAoTF1BhbG8gQWx0byBOZXR3b3JrcyxJbmMuMQ4wDAYDVQQLEwVUcmFwczEbMBkGA1UEAxMSUGFs
byBBbHRvIE5ldHdvcmtzMIIBIjANBgkqhkiG9w0BAQEFAAOCAQ8AMIIBCgKCAQEAgBhY2iXNODtcvD1pv7LVOpfMhwvPJ2jNcuKNEz2
5pni7usgYHvGq4fLD/8CYFY2tBgDEaSTZVZLabKvLOqTYM7MmT61jshA8+j2vImseWHG0dSgpgtTk696ox6XnB7QNuIeGDhtkIFqxde
tiafDWTUJ70s0RtdojegZJ7N8uOFQwmFyQTnoM/8ZrAk42mnfok8TJ/B6JE7caYPGa9Icr6FqpfFSl927O7rEUS0JeOklthLLGQUKfW
uS2FQbwgrNC5lcne81Kpi5WmeB/olkYaA2sWXVNtUZaZKUIGsjgj/irn5+moawMyk/ZLlWvmUP5W0h5KLwaotpksH8RniVKNQIDAQAB
oyEwHzAdBgNVHQ4EFgQUJdW6+XcHVipKSOhmlCh8peYHULQwDQYJKoZIhvcNAQELBQADggEBAG3YpZUE3RxDe9pak4NZZE/H8EUBTkm
yjd3xyjv7Oxo5Yn2n7Zxamyi7UIPEGH6UJGxHYagYYkUaNTFboLUYRGX6TCPbFh2klbWZNCcyx6BdDy7aDYDh63K9PZvv/BQnNc8BEA
yB8cOPyvxUL08RScJOOIj7l+Y7/CPk13OTmcEZVXIHL8X4YkuFYVx2IhagtEBp7QywheknkPiUH4uKOKKVktQISwcmsaU/+JOEF0Hxj
F1iMyiy4tnOtmRf4baiCHoGxIEPOOUvmmNaB0UosZpiG9pG/P+PjD8zxpXow+x1kj4BQSdRd1YsgxYNFePMU5HrEyNaY4XaA/XwYtZR
An4=

Skycure / SEP Mobile

MIIDYDCCAkigAwIBAgIEUt6IkzANBgkqhkiG9w0BAQUFADByMQswCQYDVQQGEwJJTDEQMA4GA1UECBMHVW5rbm93bjERMA8GA1UEBxM
IVGVsIEF2aXYxFTATBgNVBAoTDFNreWN1cmUgTHRkLjEQMA4GA1UECxMHVW5rbm93bjEVMBMGA1UEAxMMU2t5Y3VyZSBMdGQuMB4XDT
E0MDEyMTE0NDc0N1oXDTQxMDYwODE0NDc0N1owcjELMAkGA1UEBhMCSUwxEDAOBgNVBAgTB1Vua25vd24xETAPBgNVBAcTCFRlbCBBd
ml2MRUwEwYDVQQKEwxTa3ljdXJlIEx0ZC4xEDAOBgNVBAsTB1Vua25vd24xFTATBgNVBAMTDFNreWN1cmUgTHRkLjCCASIwDQYJKoZI
hvcNAQEBBQADggEPADCCAQoCggEBAJljcPl/sxfglWKF8q3h3UiANwlfMsJbX0qAspYbF1R6wuARP1dfDMXn545Gos1XO70cRN3mWE3
LZDgkD3NJhMChO3Kn1Ys3yyx977F2q6aLyRdqL8x62C12ydzo/gBscztW/1tiQCdRPDEMBkn1IUFSJ2jnBWmBXt8+m2tcPFW6mwnPEM
yknpFm7JuiYS5eXsGUPchuURS89pOA08Vqpu/r7XekhAUnFv8+BiUxOU9qlRjodW68VCItbPvNPzItQZAJai9bRcsw+gq3fkO59knM2
Mm6hyjflzWzmA3AlRwzUIFhbFevvTIngTXGufrtFWS10CZfJm+2T1aEOOOEiycCAwEAATANBgkqhkiG9w0BAQUFAAOCAQEAZldY29PR
yjMRVpnlypXvC8r7edmdjE5n64v98GMvbDdazD7YSo0y4GgACde1aVCxBrtoc2qYDB0+wpT6B+gpJrIkJI3H+qwfH7aP2ogcrRXlGdw
5Ww5MXikByjiJUhqTJ9Lak0xypht+4wJA4EXdBUOhhZAbVsoAgDuFPNbyMtd0cxljbppfO+u2B/GxpTuYRbbSIQT8cE4eu6/KSf+xI
Q//kUqlDp4quTF4aqZBrnDJZtjHVt2QWDd/+3ZT/6Pe9Wh54eQiCRntVWOB/M7GRIaP2cfMofxCmKbd+iPU2l7B8L+WnMbOsZ2c973f
eOO8RRNNd3ycVrozAlv8psxI6Q==

Cogosense / bSafeMobile

MIIDzTCCArWgAwIBAgIEIw6v3zANBgkqhw0BAQsFADCBljELMAkGA1UEBhMCQ0ExGTAXBgNVBAgTEEJyaXRpc2ggQ29sdW1iaWExEzA
RBgNVBAcTClBvcnQgTW9vZHkxIjAgBgNVBAoTGUNvZ29zZW5zZSBUZWNobm9sb2d5IEluYy4xFDASBgNVBAsTC0VuZ2luZWVyaW5nMR
0wGwYDVQQDExRDb2dvc2Vuc2UgVGVjaG5vbG9neTAeFw0xNjA1MjAwODQzMzZaFw00MzEwMDYwODQzMzZaMIGWMQswCQYDVQQGEwJDQ
TEZMBcGA1UECBMQQnJpdGlzaCBDb2x1bWJpYTETMBEGA1UEBxMKUG9ydCBNb29keTEiMCAGA1UEChMZQ29nb3NlbnNlIFRlY2hub2xv
Z3kgSW5jLjEUMBIGA1UECxMLRW5naW5lZXJpbmcxHTAbBgNVBAMTFENvZ29zZW5zZSBUZWNobm9sb2d5MIIBIjANBgkqhkiG9w0BAQE
FAAOCAQ8AMIIBCgKCAQEAmumtaA86vRbZMoxuVWCUIzLW5+FAbXq1jkGyT6SIt+wOnL0cJ6CtizgPXAaCl2A77d1tofj8I4inp+1FPe
JhSZS/UvZJK7mvWl8y0tyr1tJYtkwhYdCha/lXxkqBdAUAo++YRg08X9DF7sTaPZF6Hi30TVorYffJQ2vF2O9ofIHyAOIZ9D4GVImWg
gLJRc5FZI6iojGz0KtAxZj31HDBBbIpnTRXnfDXpuzE+jtVrbayoErSnLDp5GEtiwDv851K/3l8jj2PtdEQX1oLcAUw8Kq0lnNc1btj
2TfGw+eg1h+TLQhYMjJb1DTXDV8lPz3WbRwRJJILuLri3gLZopVgBQIDAQABoyEwHzAdBgNVHQ4EFgQUZQfKgRF7ymdJSSKe2PTtSHo
+64cwDQYJKoZIhvcNAQELBQADggEBADJvP8rKGPhc7soh5H5AE0cV7ECNeDPRGmz8JqxqXAjBOyqefyte7FjV7e28BBXtEw2POGxLwm
dSOstxAKW4QwfzuhhXWlN1PfjI948UbFlPqbB22GtWnU7urLL5OD4pbZIrK//Z4EILJFrGN4o3xRKGC369DSrLRW1ietYDp0TC/bkpt
17dcMjbpg9T3LwtRZk0vLEoCTsmOBi/Y9k37X4QRozX3oAP9zHrltA2RFJcHY8pG4RR7Mxg6B6H+ltL3EQMtK1PhPoBmWQfjeo3YxHP
OZY3DBdpT/tYm/qSom26LeUTw274KHdJ5akLMsPKdVoMPKWeyTbKK2vpRPfAXaw=

9Folders / Nine Work

MIIBqTCCARKgAwIBAgIEUlIPdDANBgkqhkiG9w0BAQUFADAYMRYwFAYDVQQHEw05Rm9sZGVycyBJbmMuMCAXDTEzMTAwNzAxMzM0MFo
YDzIwNjMwOTI1MDEzMzQwWjAYMRYwFAYDVQQHEw05Rm9sZGVycyBJbmMuMIGfMA0GCSqGSIb3DQEBAQUAA4GNADCBiQKBgQCfvpIUAr
VlzNxWaHBV3wphJILBRx0oNs/6hI7ija2Zlft443IdGsLQ/gZHCDaadPTKqh3WS2YrD4a53fGikrvFEYpGnyVVuAQgMZoHZ96fgbW2c
OlVhYFtERnNEpPgyK1pLfFnehHhQXJSP43VtCvDLLG0gD6tlefzzazZHmdWBwIDAQABMA0GCSqGSIb3DQEBBQUAA4GBAIZHTlPHb3VR
baUNXXoxIjxzbrMR+YMnxOegn+HEsBbHmCocpF7yWEAIFRdIe71XpME7ECXsdsXdwboLO/ll+IQpnPBcM6Cxl0fqhM6ry2393MOeZaz
IIpt4KOuYjQb+bZv3ghtuq5XmmgLnMNk1+Y1tKu3a4yf2Xv8/UvxOtz9I

Appthority

MIIDpTCCAo2gAwIBAgIEZnL5UjANBgkqhkiG9w0BAQsFADCBgTELMAkGA1UEBhMCVVMxEzARBgNVBAgTCkNhbGlmb3JuaWExFjAUBgN
VBAcTDVNhbiBGcmFuY2lzY28xFzAVBgNVBAoTDkFwcHRob3JpdHkgSW5jMRUwEwYDVQQLEwxBbmRyb2lkIFRlYW0xFTATBgNVBAMTDE
Fua2l0IERldmFuaTAgFw0xNjAxMDUxNjI4NDBaGA8yMjg5MTAyMDE2Mjg0MFowgYExCzAJBgNVBAYTAlVTMRMwEQYDVQQIEwpDYWxpZ
m9ybmlhMRYwFAYDVQQHEw1TYW4gRnJhbmNpc2NvMRcwFQYDVQQKEw5BcHB0aG9yaXR5IEluYzEVMBMGA1UECxMMQW5kcm9pZCBUZWFt
MRUwEwYDVQQDEwxBbmtpdCBEZXZhbmkwggEiMA0GCSqGSIb3DQEBAQUAA4IBDwAwggEKAoIBAQCiDD7a1lPf00zEOD3EgQcdsmJSHoZ
+XU+Fg2KhuuAlkC0dHNafqDaXYc4fSaWgsofYDrw7Ppx3yr+J+qLLhgfYJNX751gr++rYKECFwMAlAOL0BwoNXMpXPj5KTw9vAwa8kb
szN98Narw8rKa5tzjJpk+URSXm3fWXbuAy3D5DY54QuX/2q3PLii6kl7DV7wqJOo2r0c6IMYyBOgXagPaHMWJeXyVg4e23jLhPpoU7k
9Dv1Og5bfNLUjwEyU2dUIoZ/krN1iM1pq+iQQoYCnWVA+096kM40FA0fQ/4J4ZA4sZAsFcpr0yHEUyj3BPEKNCwkPC6A8ig7LaQeShO
pC+bAgMBAAGjITAfMB0GA1UdDgQWBBSfnTsTxAZQ8+4OjrwNGGijadgHzTANBgkqhkiG9w0BAQsFAAOCAQEAfj341f4vAXjp1KS9s7V
7/mbzK59/CI5HNaI96MLe3T/pIqbOUerijNurTCpr1Z7WRyghi70/V80otVgJn356d2FfBzwcsfkYw36f44mUOEolo2XQuE28ElWLiq
E7uqClyFTbD9+2QULvSpb4/V+7hygNq1faX97OoSpJFXz5igE6e8zaKc+81AuTEnHLiJHWo6o1TjAerMEnz+eIwejt8q8VBcbQ6t4SD
YJMhm+u/edOKiQBNxJ13/3KWdGqc4PoQroFwcJahKhjUZhDAu+3P+zCWqlLY4NYANSeN4LPV29sSv7ZK5ZEsOh8XOW6P/JqLlgPOuDx
45L2UTm3QrFUXQ==

Movius

```
MIIDjjCCAnagAwIBAgIEUKqV8jANBgkqhkiG9w0BAQUFADCBiDELMAkGA1UEBhMCVVMxEDAOBgNVBAgTB0dlb3JnaWExFDASBgNVBAc
TC0pvaG5zIENyZWVrMScwJQYDVQQKEx5Nb3ZpdXMgSW50ZXJhY3RpdmUgQ29ycG9yYXRpb24xEjAQBgNVBAsTCU1hcmtldGluZzEUMB
IGA1UEAxMLSm9zZSBSb21lcm8wHhcNMTIxMTE5MjAyNjI2WhcNNDAwNDA2MjAyNjI2WjCBiDELMAkGA1UEBhMCVVMxEDAOBgNVBAgTB
0dlb3JnaWExFDASBgNVBAcTC0pvaG5zIENyZWVrMScwJQYDVQQKEx5Nb3ZpdXMgSW50ZXJhY3RpdmUgQ29ycG9yYXRpb24xEjAQBgNV
BAsTCU1hcmtldGluZzEUMBIGA1UEAxMLSm9zZSBSb21lcm8wggEiMA0GCSqGSIb3DQEBAQUAA4IBDwAwggEKAoIBAQDiigQgehRMGOU
KfZILdgHonkkAOmySVFOkzNO7Fv+oMWB+qVvlj2ZQdNqyiTkj5nhxGjArj5MobDTc9zH42anpYcfU7KyMj6vpMFZfpyfROUyFXdTWEL
0b3lXd1QZAZwzBg5yDmklNqHHJ1z8moA18PLg+X0b2W92CFxbkkacG1zJnZ6GjJkfMPfQ/sQcayK+hXdr3CaavduDUsIdQoa3IfkiLG
Hw62zHmOhcPeV/lwDwhe+lXPew/VNapOuBZgNPnm22pLYrpPW5MMk2Oq3lcOspCZJDXlp/zfhiE4xH91CIsKBStP5c6buQJd9cwrPpq
X+fobDOzdOeAZt5Z9E25AgMBAAEwDQYJKoZIhvcNAQEFBQADggEBAK72k97hFpHHhuTfJiRAAy/TsU54MyDawvVMCqTd/kQDAODDA/j
QDv6U380wIG0uB5rn5UyoKbH4ZN0hizhiAAdIsIBR9B97mnl9zZGlk2x4sxECHjs/YcXkyLL8lcALVMoph+WGCPR7RzPlWbvqFOFecq
k/cCNcNGLVI2Qm/TuKMvCvdIjGlRgY2Car9w3NbQ7y4Avflv2mzuzHI4zLaxgGs7n6Jw0jiDOruzL1NbiuldYABkM+XCLtNtTTqpeN3
2dFrZOH6d4cJkAjTqh2iYd+HcYDX3t6N9pA9ADAunKJbfOMgjW0rrcvSc6VkhpMH4Tk2WyAdYdZCpFkSWQnt+M=
```

NetDocuments

```
MIICzTCCAbWgAwIBAgIEXS/TXzANBgkqhkiG9w0BAQsFADAXMRUwEwYDVQQKEwxOZXREb2N1bWVudHMwHhcNMTcwNjI2MTAwOTQ1Whc
NNDIwNjIwMTAwOTQ1WjAXMRUwEwYDVQQKEwxOZXREb2N1bWVudHMwggEiMA0GCSqGSIb3DQEBAQUAA4IBDwAwggEKAoIBAQDTSWa+Gj
VADWgH3Wj+ZGeoxVEcY4uFZTIR6f9MWjEGUNNPHf9Ljs8CUcvlu6xXmWOtrAP8U/++0kkq7izkanT9of3T6bS/4DKLVeOzISg42xQzo
dGIz+zZhQ2+2F7x2KlV59F7IthZPbRTn86D2YhKs6YM0aFotMnX13lpWpdPEnuN4n7CKsfKwbj4VHMkMUTAjpml+TWLtVx55BoY+XLx
r26G9/osp6Q3Hr8iHGFwCop72tE5vpdMyhD7+zuW0VueFriijpGFnr5R5zTHAlgBks9HFW5tjbnXIBp44ytzFE8ju1aQ/zSZy9SgnAd
JS6HEHnuwV18BgBZEDIJZJXNpAgMBAAGjITAfMB0GA1UdDgQWBBTyt4Alh+gNKyyYvk5dHy9A4PVv4TANBgkqhkiG9w0BAQsFAAOCAQ
EAs2DCK/12wC0bwoAECpZWfEfTorZFyM+6EkrhdY8WNdztZCyIBqfBTrtbhXQVRQ7gZta8mfoaUoNma2ZXYdzZ3RHnkY+4jVdPgIJPw
lBFjiuiQlOayoc/Bqu25OAJ2f2j0ygfB/5ncg5BHuPBQRMzUXTi60/43WkOMf0MwBCBoTwwaHoZTaLCaoAOh2kkIIDnoaTN0paxC2r7
Hkcr1foui4LKsO7hOM6oO8C8fJXBZdO320+NtfrXSCU0hnHT8xbYLUwqWLqb0LtOTnFZbtuWEdES38b7a06oQxd1ngYCAXNhpsd8J3A
fykBhMCGoWFLUcshHIH1W/oD73QD7pd+UlA==
```

IBM

```
MIICqDCCAmagAwIBAgIETNrtMjALBgcqhkjOOAQDBQAwNzELMAkGA1UEBhMCVVMxDjAMBgNVBAsTBUxvdHVzMRgwFgYDVQQDEw9JQk0
gQ29ycG9yYXRpb24wHhcNMTAxMTEwMTkwNjI2WhcNMzgwMzI4MTkwNjI2WjA3MQswCQYDVQQGEwJVUzEOMAwGA1UECxMFTG90dXMxGD
AWBgNVBAMTD0lCTSBDb3Jwb3JhdGlvbjCCAbgwggEsBgcqhkjOOAQBMIIBHwKBgQD9f1OBHXUSKVLfSpwu7OTn9hG3UjzvRADDHj+At
lEmaUVdQCJR+1k9jVj6v8X1ujD2y5tVbNeBO4AdNG/yZmC3a5lQpaSfn+gEexAiwk+7qdf+t8Yb+DtX58aophUPBPuD9tPFHsMCNVQT
WhaRMvZ1864rYdcq7/IiAxmd0UgBxwIVAJdgUI8VIwvMspK5gqLrhAvwWBz1AoGBAPfhoIXWmz3ey7yrXDa4V7l5lK+7+jrqgvlXTAs
9B4JnUVlXjrrUWU/mcQcQgYC0SRZxI+hMKBYTt88JMozIpuE8FnqLVHyNKOCjrh4rs6Z1kW6jfwv6ITVi8ftiegEkO8yk8b6oUZCJqI
Pf4VrlnwaSi2ZegHtVJWQBTDv+z0kqA4GFAAKBgQCpzsGYWkP3Ab7mgl8gdEGnBH91U+6R/3UJ0lB5AyRlokcuTwzA7aMhj34tU7x2S
grBhwYYfGmlsoNx5Ad41Qr4KwAuY9/6wlgKrHSVZoO/xVHwT4pyjof/W+MC7n9FacDz+kvVBCWAXqyIMG5QK/u3dTT6ew/qH0WVGwt4
JBTPFDALBgcqhkjOOAQDBQADLwAwLAIUILyh9fOFfHEOtYvYYnGVbAVhCg0CFCoLbVTdLhFwn6H1i427vvKu4Qsh
```

TrustBridge

```
MIICtzCCAZ+gAwIBAgIEJ7uJNDANBgkqhkiG9w0BAQsFADAMMQowCAYDVQQDEwFLMB4XDTE4MDUxNTE1NTc1NloXDTQzMDUwOTE1NTc
1NlowDDEKMAgGA1UEAxMBSzCCASIwDQYJKoZIhvcNAQEBBQADggEPADCCAQoCggEBALSNBYa0peFLpIIyxIWsy9iQ2ILwC2hACQ9lEi
pXL1OBKyZxym5HUGnHX0/W6XG4OTAfMmQ10WA3skJB+YCpD2fD8wZMNEJGiHmf/wcrEzxtC8sVcJm2Cw38/l8fgryep6LTb/dnbTV35
wN4+25X+W6kxaK200AYhqfDvtBoNZUu85OWxLwJ2LzXHUdfme0SjHcdaOhI/FZGLoGLAzVDpxSGm3WuMvWr61hrcoVuR0Iq7CP8pa0w
KTul92GJ+erTiqBGfjFhFS7tD4asU/qo5G5+Eb6+Tha/agz+k36bOOy6+8HJwmsaD7CzVdBvRkVeu+tczWSOQfl1OPGUTY0yXeECAwE
AAaMhMB8wHQYDVR0OBBYEFPAl539ff3Ymbf+sBzeVuKRltJR8MA0GCSqGSIb3DQEBCwUAA4IBAQBm0NlVraBd6Gak/Jg0P3kIYc7l0x
TnSrmb/994RAXBOi1eQGkzwfy3nEe4cn4IkFIlvpyECZBPF04qlCd8BsNFLMsMJQZH64/IG0mfOeNCrZmR702rwK1jeX9CqAicgYgj/
O9aBZ3G4EhKhijbQ4DBZ/Pw9hu2xabhOwnCbRifpCcwmOC4JGH/Hl3okAKU3k1/lSL5TW4LRUbT0hHrV0YEpe9rfbS03xsyzhjaw/r9
sRfb6POzk8HvZyU99T7uXG1RvxEdBbHvZWx4OyRcFt1fCQDSur17x3VIqKFYvqGr/cK+o7Nd10BLE9GVzdxkoXf/XP/V7/xlbfR+588
7uJrD
```

SDK Profile Refresh Interval

The SDK profile applies SDK functionality to an SDK-built app. The SDK retrieves updates from the console at varying intervals based on the app's activity status.

Purpose of the SDK Profile

You enable an SDK-built app with SDK functionality using code and settings in the Workspace ONE UEM console. The SDK profile is what carries the console configurations to the application.

You configure default profiles in the Groups & Settings > All Settings > Apps > Settings and Policies and then select from Security Policies, Settings, or SDK App Compliance.

You configure custom profiles in Groups & Settings > All Settings > Apps > Settings and Policies > Profiles.

After you configure the profile, you assign it to the application when uploading it to the console. The refresh interval is the schedule the Workspace ONE UEM console sends changes and updates to the SDK profile assigned to the app.

Refresh Interval When App is in the Background

The SDK updates the SDK default or custom profile when it has been 30 minutes since the last profile retrieval for these app activity statuses.

- The app transitions from the background to the foreground.
- The app wakes from doze mode.

You can change the time to retrieval with the API AWSDKApplicationDelegate.getScheduleSdkFetchTime().

Refresh Interval When App is in the Foreground

The SDK updates the SDK default or custom profile when it has been four hours since the last profile retrieval for this app activity status.

- The app is in the foreground and remains there for an extend period.

You cannot change this value.

Run a Process Before Initialization, Optional

Use this optional procedure to run processes that analyze the environment into which the application is deployed. For example, run a process to determine if your application needs to start the SDK in a specific environment. To run a process before initialization in your SDK-built application, edit the AndroidManifest.xml file and customize an activity.

Procedure

1. In the AndroidManifest.xml file, remove the launcher tag `<category android:name="android.intent.category.LAUNCHER" />`.

   You add the tag in the placeholder activity you create to run your process.

2. Create an activity and register it in the AndroidManifest.xml file. This activity runs the desired process.

3. Add the intent filter to the activity you just created in the manifest. The filter resembles the example.

```
<intent-filter>
            <action android:name="android.intent.action.MAIN" />
            <category android:name="android.intent.category.LAUNCHER" />
</intent-filter>
```

4. Call startActivity(new Intent(this, SDKSplashActivity.class)) after the process completes.

Certificate Pinning

The Workspace ONE software development kit (SDK) for Android has a number of certificate pinning customization features. Some examples follow.

1. Use the programming interfaces:
   - `getScheduleSdkFetchTime()`
     Override this method to change when the login module fetches updates to SDK settings from the Workspace ONE UEM console.
   - `getKeyManager()`
     Override this method to a value rather than null so that the login module initializes another key manager and not its own.
2. In the AndroidManifest, add the certificate pinning meta tags under the opening application tag.

```
<application
    ... attributes here ...>
    <meta-data
        android:name="com.airwatch.certpinning.refresh.interval"
        android:value="1"/>
    <meta-data
        android:name="com.airwatch.certpinning.refresh.interval.unit"
        android:value="DAYS"/>

    ...

</application>
```

Base Activity

If you need the SDK authentication, DLP, and timeout behavior, app activities should extend from SDKBaseActivity. These activities allow the application to handle the life cycle correctly and to manage the state of the SDK.

If the app Activity class extends SDKBaseActivity, then the SDK will enforce any data loss prevention (DLP) restrictions specified in the management console. Some examples of actions that can be restricted by DLP are as follows.

- Copy and paste out of the app.
- Copy and paste into the app.
- Application data backup.
- Screen capture, also known as screenshot.

The SDKContext and Custom Data Encryption

To encrypt custom data and to secure storage data, use the data encryption API set to encrypt and decrypt data.

Retrieve the SDK Profile

Once the SDKContext is in the configured state, you can call the SDKContext getSDKConfiguration() method to retrieve the SDK profile. The SDK must be finished with its configuration otherwise calling getSDKConfiguration() returns with an empty value.

```
if (sdkContext.getCurrentState() == SDKContext.State.CONFIGURED) {
    String sdkProfileString = SDKContextManager.getSDKContext().
    getSDKConfiguration().toString();
}
```

Encrypt Custom Data

Once the SDKContext is in the initialized state, you can call the data encryption API set. This set of functions uses the Workspace ONE SDK's intrinsic key management framework to encrypt and decrypt any data you feed in.

Use the MasterKeyManager API set when the SDK is in an initialized or configured state. See the example of how you can encrypt and decrypt a string value.

```
if (SDKContextManager.getSDKContext().getCurrentState()!=SDKContext.State.IDLE) {
    MasterKeyManager masterKeyManager = SDKContextManager.getSDKContext().getKeyManager();
    String encryptedString = masterKeyManager.encryptAndEncodeString("HelloWorld");
    String decryptedString = masterKeyManager.decodeAndDecryptString(encryptedString);
}
```

Secure Storage Data

After the SDKContext is in the initialized state, you can call the secure storage API set. This set of functions stores key value pairs in encrypted storage.

```
if (SDKContextManager.getSDKContext().getCurrentState()!=SDKContext.State.IDLE) {
    SecurePreferences pref = SDKContextManager.getSDKContext().getSDKSecurePreferences();
    pref.edit().putString(<KEY_NAME>, <VALUE>).commit(); // to store value
    Object value = pref.getString(<KEY_NAME>, <Default_Value>);
}
```

APIs for Copy and Paste Restrictions

To use the Workspace ONE SDK copy restriction, replace the Android classes in your application to the listed Workspace ONE APIs.

Examples

If Java class XYZ extends `TextView{...}`, change it to extend `AWTextView{...}`.

If you override the method onTextContextMenuItem(int id), do not process the listed IDs. You must call return super.onTextContextMenuItem(id); for the listed IDs.

- android.R.id.cut
- android.R.id.copy
- android.R.id.paste
- android.R.id.shareText

Change `<TextView>` in all Layout XML or View XML to `<com.airwatch.ui.widget.AWTextView.../>`.

APIs

Android Class	Workspace ONE SDK API
android.support.v7.widget.	com.airwatch.ui.widget.
AppCompatEditText	AWEditText
android.support.v7.widget.	com.airwatch.ui.widget.
AppCompatTextView	AWTextView
android.support.v7.widget.	com.airwatch.ui.widget.
AppCompatAutoCompleteTextView	AWAutoCompleteTextView
android.support.design.widget.	com.airwatch.ui.widget.
TextInputEditText	AWTextInputEditText
android.support.v7.widget.	com.airwatch.ui.widget.
SearchView.SearchAutoComplete	AWSearchAutoComplete
android.webkit.WebView	com.airwatch.ui.widget.CopyEnabledWebView

Push Notification Support

Code Apps to Use the SDK to Send Push Notifications with FCM

If you use Firebase Cloud Messaging (FCM), use two Workspace ONE SDK for Android APIs in your SDK-built, internal application to integrate your FCM solution with Workspace ONE UEM. This feature replaces sending push notifications with GCM.

The Workspace ONE SDK for Android handles two tasks with APIs.

- Accept the registration token and send it to the console.

```
/**
* API to register the Notification Token with the WS1 Console.
*/
PushNotificationManager::registerToken(token: String)
```

- Check for and recieve notifications from the application based on the notification parameters.

```
/**
* API to notify the SDK that a message has been received.
*
* @param title: Notification Title
* @param body: Notification Message
* @param payload: (Optional) Generic container to downstream multiple actions, data, etc.
*/
PushNotificationManager::processMessage(
    title: String, body: String, payload: Map<String, Any>?)
```

Prerequisites for FCM

Devices must have an internal, SDK-built application deployed to them that supports push notifications for this to work. Also, your application needs to handle getting the registration token and the sending of push notifications not delegated to the SDK. The Workspace ONE SDK for Android does not handle token registration and non-delegated push notifications for the FCM model.

Procedure for FCM

1. Use the register token API to register the FCM token with the SDK.

```
FirebaseInstanceId.getInstance().getInstanceId().addOnCompleteListener(task -> {
            if (task.getResult() != null && !TextUtils.isEmpty(task.getResult().getToken())) {
                String token = task.getResult().getToken();
                regId.setText(String.format(getString(R.string.registration_id), token));

PushNotificationManager.getInstance(this.getApplicationContext()).registerToken(token);
            } else if (task.getException() != null) {
                regId.setText(String.format(getString(R.string.error_while_registering),
task.getException().getMessage()));
            }
        }
});
```

   Workspace ONE SDK for Android sends the token to the Workspace ONE UEM console.

2. Use the process message API to define parameters for the Workspace ONE SDK for Android to consume the push notification. Currently, the SDK checks for command actions.

```
class FCMMessagingService : FirebaseMessagingService() {
    private val TAG = "FCMMessagingService"
    override fun onMessageReceived(message: RemoteMessage?) {
        val context = SDKContextManager.getSDKContext().context

PushNotificationManager.getInstance(context).processMessage(message?.notification?.title?: "",
                message?.notification?.body ?: "You got a message", null)
    }


    override fun onNewToken(token: String?) {
        Logger.d(TAG, "onNewToken() called with $token")
        super.onNewToken(token)


        val context = SDKContextManager.getSDKContext().context
        if (token != null)
            PushNotificationManager.getInstance(context).registerToken(token)
    }
}
```

Configure Push Notifications in the Console

Use internal, SDK-built applications to send push notifications integrated with systems like FCM.

Procedure

1. Upload the application to Workspace ONE UEM.
2. Navigate to Apps & Books > Applications > Native > Internal and select the application from the list view.
3. Select the Devices tab.
4. Select all the devices to which you want to send the notification.
5. Select Send Message to All or Send.
6. Select Push Notification for the Message Type.
7. Select the SDK-built application in the Application Name field the system uses to send push notifications to selected devices.
8. Enter the message in the Message Body, complete the rest of the procedure, and send it to devices.

Send Push Notifications Through Internal Applications with GCM and Workspace ONE UEM

If you use Google Cloud Messaging (GCM), you can send push notifications from applications uploaded to the Workspace ONE UEM console to Android devices through GCM. Devices must have an SDK-built application on it that supports push notifications.

Firebase Cloud Messaging (FCM) replaces GCM. This outlined process does not work for applications that use FCM.

Prerequisites for GCM

Google has deprecated GCM as of April 10, 2018. See Code Apps to Use the SDK to Send Push Notifications with FCM.

This feature requires Workspace ONE UEM console v9.5+.

Procedure for GCM

1. To send notifications using GCM, configure GCM and get a sender ID and an API server key.

   - Sender ID
     The GCM system generates this value when you configure an GCM API project. The system uses the value to identify the package ID for the application.
   - API Server Key
     This key gets saved on application server to authorize the server to access Google services. Enter this value when you upload the internal application to the Workspace ONE UEM console.

   These two values enable GCM, Workspace ONE UEM, the client, and the application to communicate.

   The Client Gets a Registration Token from GCM. Call the registration API on the client to register with GCM. The client then sends the registration token to the Workspace ONE UEM console. Each client has its own registration token. The console uses the registration token to identify the client for which push notifications are sent through GCM servers.

2. Register the application with GCM using the directions at https://developers.google.com/cloudmessaging/android/client.
3. Retrieve the sender ID and code it in the application class.
4. Retrieve the API server key and enter it in the console while uploading the app.
5. Call Register API on the client side to register the device with the GCM server and to retrieve the registration token (GCM token).

6. The SDK beacon sends the token to the Workspace ONE UEM console. If you want to force sending the token immediately, stop and restart the application.
7. Upload the Application to Workspace ONE UEM.
8. Send push notifications with Workspace ONE UEM from the Workspace ONE UEM console.
    1. Navigate to Apps & Books > Applications > Native > Internal and select the application from the list view.
    2. Select the Devices tab.
    3. Select all the devices to which you want to send the notification.
    4. Select Send Message to All or Send.
    5. Select Push Notification for the Message Type.
    6. Select the SDK-built application in the Application Name field the system uses to send push notifications to selected devices.
    7. Enter the message in the Message Body, complete the rest of the procedure, and send it to devices.

Code GCM Push Notifications in the Application

To use push notifications in your application and Workspace ONE UEM, code the support of push notifications in the AWFramework module in the application class.

Use the PushNotificationContext Interface.

The AWFramework module provides an abstract AWApplication class that runs the PushNotificationContext with default behaviors. Your application can override some of the methods to intercept callbacks for additional actions or if it does not use the AWApplication class.

Follow this procedure to code support for push notifications if your application does not use the AWApplication class.

Prerequisites

Google has deprecated GCM as of April 10, 2018. See Code Apps to Use the SDK to Send Push Notifications with FCM.

- Code the sender ID from GCM into the application class for push notifications to work.
- Add the API server key from GCM to the Workspace ONE UEM console when the admin uploads the application.

Procedure for GCM Push Notifications in the Application

1. Override the registerForPushNotifications() method in the PushNotificationContext class and call `GCMManager.registerForPushNotifications(getApplicationContext());`.

   A call to registerForPushNotifications() initiates the GCM registration process.

2. Override getSenderID() and return your application's corresponding sender ID.
3. To intercept GCM registration and GCM push notification events, use the callbacks in IPushNotificationReceiver.
4. If your application sends other push notifications besides those for Workspace ONE UEM, override AWPushNotificationReceiver implementation and call super.

   For details about specific Workspace ONE UEM actions, review AWPushNotificationReceiver. Actions include uploading the registration token and checking for secure messages in the console. If the application class overrides AWApplication and does not use push notifications for anything else, do not override getPushNotificationReceiver().

5. To code that the application received a notification and to not use AWApplication's default behavior, override onSecureMessageReceived(String message) with the desired behavior.

   The default behavior is to display a notification in the notification bar with a Workspace ONE UEM icon.

APIs to Use Custom Certificates for Your SDK-Built Apps

The Workspace ONE SDK for Android provides APIs to retrieve custom certificate authority (CA) certificates configured in the SDK profile and to perform trust validations based on these custom CA certificates. The admin configures trusted certificates as Credentials in the SDK profile. When the SDK fetches the SDK profile, it also fetches and stores the CA certificates.

API to Retrieve CA Certificates

```
List<X509Certificate> caCerts =
SDKContextManager.getSDKContext().getKeyStore().getCACertificates();
```

API to Perform Trust Validations

Use the previous API with the following SDK provided API to perform trust validations on any X509 certificate.

The declaration is in the CertificateUtils class:

```
public boolean isTrusted(
    X509Certificate[] inputCerts, List<X509Certificate> trustedCerts
)
```

The parameters are as follows.

- `inputCerts`
  List of the `X509Certificate` chain that must be validated for trust.
- `trustedCerts`
  List of CA certificates to compare against.

If the `inputCerts` chain, up to any of the CA certificates, is `trustCerts`, then this API returns true.

Build Custom X509TrustManager Objects

You can build custom X509TrustManager objects with the two APIs. You can use them with URLConnection, OkHttpClient, and any other networking libraries that require TLS/SSL trust.

```
X509TrustManager customTrustManager = new X509TrustManager() {

    @Override
    public void checkServerTrusted(
        X509Certificate[] chain, String authType
    ) throws CertificateException
    {

        List<X509Certificate> caCerts =
        SDKContextManager.getSDKContext().getKeyStore().getCACertificates();

        if(!new CertificateUtils().isTrusted(chain, caCerts)) {
            throw new CertificateException("Certificate chain not trusted");
        }
    }
};
```

API to Perform Certificate Classification

The SDK provides an API to retrieve certificates based on the certificate use. It classifies all the certificates configured in the SDK profile into four types.

- Authentication
- CA
- Signing
- Encryption

The API classifies based on the Key Usage and Extended Key Usage attributes present in the certificates.

The following code snippet illustrates usage.

```
KeyStore authCerts = SDKContextManager.getSDKContext().getKeyStore().getKeyStoreByUsage(
    SDKKeyStore.CertificateUsage.AUTHENTICATION_CERTIFICATE);
```

This API returns a java.security.KeyStore object that contains all key-pair entries classified as authentication certificates.

Set Inactivity Timer Reset for Application Inactivity

To use the Application Inactivity setting in the default Workspace ONE SDK for Android profiles in the Workspace ONE UEM console, code your SDK-built app to reset the user inactivity timer. Code this feature with the AWFramework library.

The SDK resets the user inactivity timer implicitly whenever the application comes to the device foreground. An explicit reset is required only when user interaction takes place without the application coming to the device foreground.

Workspace ONE UEM console Configurations

Have your Workspace ONE UEM admin enable the Application Inactivity setting in the console. The admin can find this setting in Groups & Settings > All Settings > Apps > Settings and Policies > SDK App Compliance.

Code Additions

To integrate the inactivity timer reset into a Workspace ONE SDK for Android-built app, add the listed methods.

- Implement `AirWatchSDKIntentService::onClearAppDataCommandReceived`. Your app must implement wipe callbacks, that are in the ClientSDK library.
- By default, the app in the foreground is considered App Activity. Otherwise, it is considered App Inactivity. If the user can interact with the app without bringing it to the foreground, and if you want to count it as App Activity, call `ApplicationLifecycleUtil::resetInteractionTime` to reset inactivity time.

How to Implement a Wipe Callback

Here is an example of implementing a wipe callback.

```
override fun onClearAppDataCommandReceived(context: Context, reasonCode: ClearReasonCode) {
// This method is invoked by the SDK to wipe the application data, for example
// following a failed inactivity check.
//
// Run application code to do the following from here:
//
// - Delete any data that was encrypted with a key that is being deleted.
// - Delete any data that wasn't encrypted.
// - Delete any data that was encrypted with a key from elsewhere.

// The following SDK method must be called at some point in the implementation.
SDKContextManager.getSDKContext().sdkClearAction.clear(SDKClearAction.Type.ALL)
}
```

Document Information

Revision History

23jul2020	First publication, for 20.7 SDK for Android.
09sep2020	Added note about DLP enforcement
11oct2020	Updated for 20.9 SDK for Android.
18Feb2025	Brand Revision

Legal