# Key Rotation on Android
with MediaCodec/MediaCrypto/MediaDrm

Implementing Key Rotation, where successive license requests are issued to acquire the rotating key sets during playback of a live channel or VOD movie, is complicated on android devices by the fact that the MediaCodec instance is tied to a single MediaDrm session. Each MediaDrm session can only send one initial license request in its lifetime, so it appears that making successive key requests is not possible without tearing down the MediaCodec and creating a new one, which would cause playback discontinuities in the stream.
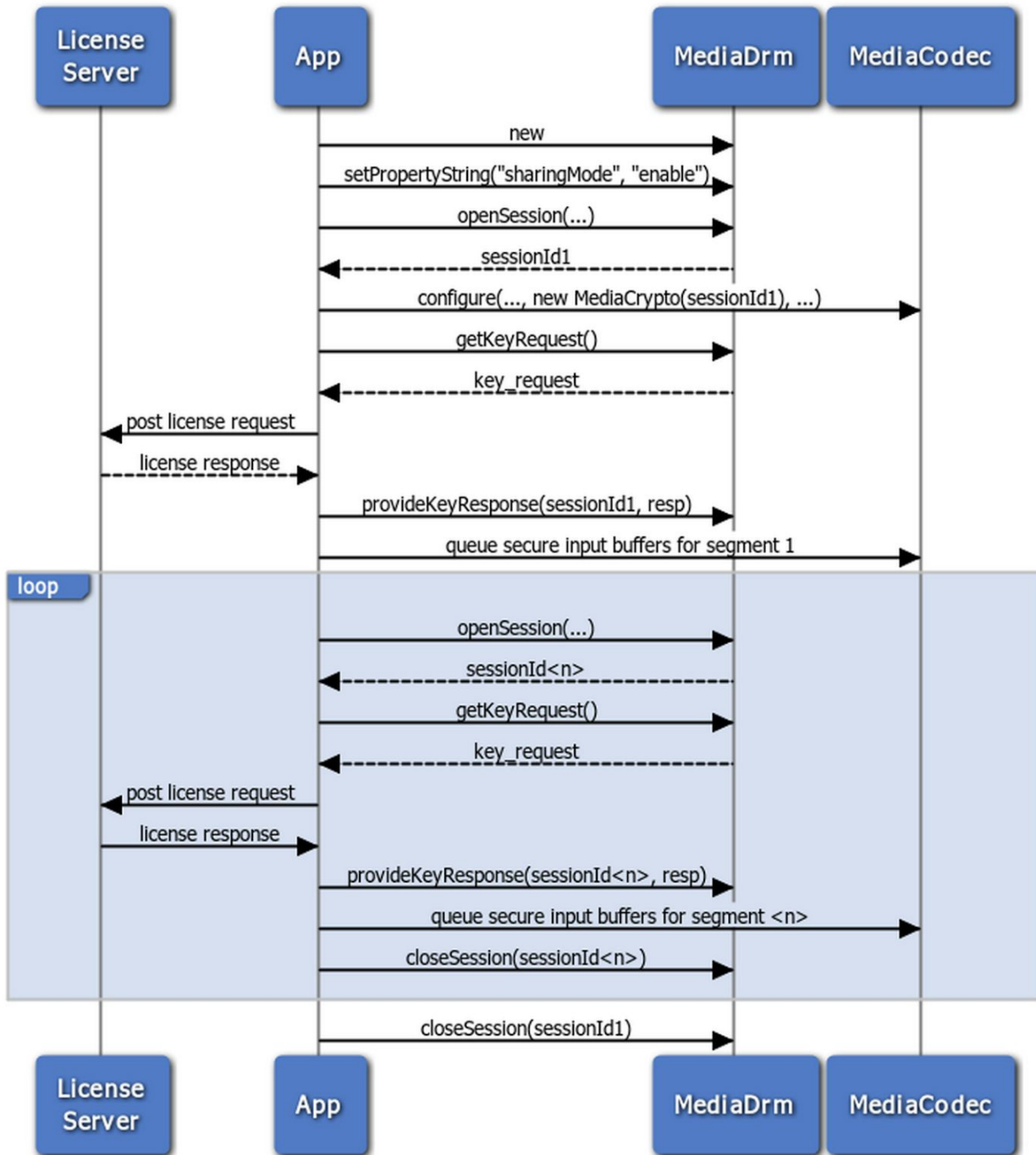

Key Rotation
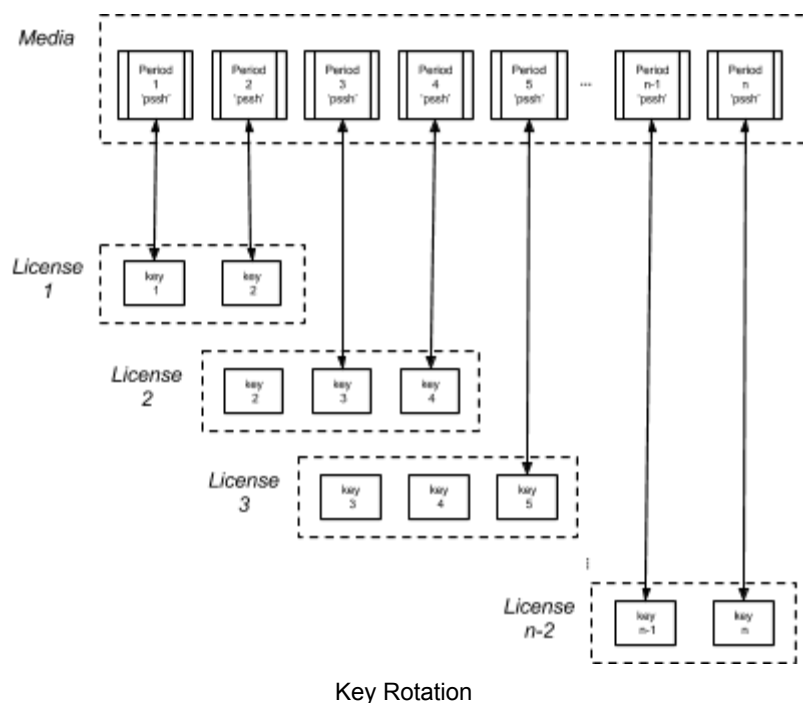
An app can avoid this limitation using the Widevine DRM plugin's session sharing mode. Session sharing mode allows all sessions created by a MediaDrm instance to share a common pool of keys/key-ids. Session sharing mode is enabled on a MediaDrm instance using:

```
MediaDrm md;
md.setPropertyString("sessionSharing", "enable");
```

Once session sharing mode is enabled, all sessions created from that MediaDrm instance will be able to access keys loaded into other sessions, effectively creating one common pool of keys. The sequence diagram below shows how the app can use session sharing mode to implement key rotation.

Key Rotation with MediaDrm

License Server — App — MediaDrm — MediaCodec

App → MediaDrm: new

App → MediaDrm: setPropertyString("sharingMode", "enable")

App → MediaDrm: openSession(...)

MediaDrm ⇢ App: sessionId1

App → MediaCodec: configure(..., new MediaCrypto(sessionId1), ...)

App → MediaDrm: getKeyRequest()

MediaDrm ⇢ App: key_request

App → License Server: post license request

License Server ⇢ App: license response

App → MediaDrm: provideKeyResponse(sessionId1, resp)

App → MediaCodec: queue secure input buffers for segment 1

loop

App → MediaDrm: openSession(...)

MediaDrm ⇢ App: sessionId<n>

App → MediaDrm: getKeyRequest()

MediaDrm ⇢ App: key_request

App → License Server: post license request

License Server → App: license response

App → MediaDrm: provideKeyResponse(sessionId<n>, resp)

App → MediaCodec: queue secure input buffers for segment <n>

App → MediaDrm: closeSession(sessionId<n>)

App → MediaDrm: closeSession(sessionId1)

Note that the original session (*sessionId1* in the sequence diagram) cannot be closed until playback is complete, since that is the session registered with MediaCodec/MediaCrypto.

In recent versions of android (Marshmallow and newer), this process has been simplified through the addition of a MediaCrypto.setMediaDrmSession(sessionId) method, so session sharing mode will not be necessary.