Proprietary Rights
The information contained in this document is proprietary and confidential to Mastercard International Incorporated, one or more of its affiliated entities (collectively "Mastercard"), or both.

This material may not be duplicated, published, or disclosed, in whole or in part, without the prior written permission of Mastercard.

Trademarks
Trademark notices and symbols used in this document reflect the registration status of Mastercard trademarks in the United States. Please consult with the Customer Operations Services team or the Mastercard Law Department for the registration status of particular product, program, or service names outside the United States.

All third-party product and service names are trademarks or registered trademarks of their respective owners.

Disclaimer
Mastercard makes no representations or warranties of any kind, express or implied, with respect to the contents of this document. Without limitation, Mastercard specifically disclaims all representations and warranties with respect to this document and any intellectual property rights subsisting therein or any part thereof, including but not limited to any and all implied warranties of title, non-infringement, or suitability for any purpose (whether or not Mastercard has been advised, has reason to know, or is otherwise in fact aware of any information) or achievement of any particular result. Without limitation, Mastercard specifically disclaims all representations and warranties that any practice or implementation of this document will not infringe any third party patents, copyrights, trade secrets or other rights.
# Table of Contents

**Chapter 1  About this Document** .................................................................................. 4  
Purpose ................................................................................................................................. 4  
Audience ............................................................................................................................... 4  
References ............................................................................................................................. 4  
Abbreviations ...................................................................................................................... 4  

**Chapter 2  Introduction** ............................................................................................... 6  
Architecture .......................................................................................................................... 6  
MP SDK Responsibilities ..................................................................................................... 8  
Development Support ......................................................................................................... 8  
  Dependencies .................................................................................................................... 9  
  Integration ......................................................................................................................... 9  
Minimal Permissions Required ............................................................................................ 9  
Third Party Libraries .......................................................................................................... 9  

**Chapter 3  Mobile Payment SDK Assets** .................................................................... 10  

**Chapter 4  Security Countermeasures** ...................................................................... 13  
Communication ................................................................................................................ 13  
  Transport Layer ............................................................................................................... 13  
  Application Layer ........................................................................................................... 13  
Storage Protection ............................................................................................................. 13  
Binary Protection .............................................................................................................. 14  
Operating System Specific ............................................................................................... 14  

**Chapter 5  Guidance** ................................................................................................. 15  
Additional Security Measures .......................................................................................... 16
Chapter 1   About this Document

Purpose

The purpose of this document is to:

- Provide security guidelines to MPA Developers (as defined below) using the Mastercard MP SDK.
- Provide security evaluation labs with the necessary information on the security gaps and goals covered by the Mastercard MP SDK.

Audience

This document is intended for developers licensed by Mastercard to use the Mastercard MP SDK to develop a mobile payment application (MPA) for contactless and/or remote payment (MPA Developers).

References

The following publications contain information directly related to this document or are referenced by it. The latest version (and associated Errata) applies unless a publication date is explicitly stated.

[1]. Mastercard Cloud-Based Payments – Mobile Payment Application Functional Description. Version 1.0 – August 2014

[2]. MPA SDK Developer Guide – Guidance to Developers to Integrate the MCBP product into their mobile application


Abbreviations

The following abbreviations are used in the document.

<table>
<thead>
<tr>
<th>Acronym</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>API</td>
<td>Application Programming Interface</td>
</tr>
<tr>
<td>ATC</td>
<td>Application Transaction Counter</td>
</tr>
<tr>
<td>CMS</td>
<td>Credential Management System</td>
</tr>
<tr>
<td>Acronym</td>
<td>Description</td>
</tr>
<tr>
<td>---------</td>
<td>--------------------------------------</td>
</tr>
<tr>
<td>CMS-D</td>
<td>Credential Management System - Dedicated</td>
</tr>
<tr>
<td>CVM</td>
<td>Cardholder Verification Method</td>
</tr>
<tr>
<td>DEK</td>
<td>Data Encryption Key</td>
</tr>
<tr>
<td>HCE</td>
<td>Host Card Emulation</td>
</tr>
<tr>
<td>HTTP</td>
<td>Hyper-Text Transfer Protocol</td>
</tr>
<tr>
<td>ICC</td>
<td>Integrated Circuit Card</td>
</tr>
<tr>
<td>IDE</td>
<td>Integrated Development Environment</td>
</tr>
<tr>
<td>IDN</td>
<td>ICC Dynamic Number</td>
</tr>
<tr>
<td>LDE</td>
<td>Local Database Encrypted</td>
</tr>
<tr>
<td>MAC</td>
<td>Message Authentication Code</td>
</tr>
<tr>
<td>MDES</td>
<td>Mastercard Digital Enablement System</td>
</tr>
<tr>
<td>MP SDK</td>
<td>Mobile Payment Software Development Kit</td>
</tr>
<tr>
<td>MPA</td>
<td>Mobile Payment Application</td>
</tr>
<tr>
<td>MPP</td>
<td>Mobile PayPass</td>
</tr>
<tr>
<td>NFC</td>
<td>Near-Field Communication</td>
</tr>
<tr>
<td>PIN</td>
<td>Personal Identification Number</td>
</tr>
<tr>
<td>SSL</td>
<td>Secure Socket Layer</td>
</tr>
<tr>
<td>RNS</td>
<td>Remote Notification System</td>
</tr>
<tr>
<td>TLS</td>
<td>Transport Layer Security</td>
</tr>
<tr>
<td>TSM</td>
<td>Trusted Service Manager</td>
</tr>
<tr>
<td>UI</td>
<td>User Interface</td>
</tr>
<tr>
<td>WBC</td>
<td>White-Box Cryptography</td>
</tr>
</tbody>
</table>
Chapter 2    Introduction

The contents of this document correspond to Mastercard MP SDK version 1.0.6 (including revisions 1.0.6a, 1.0.6b, 1.0.6c and 1.0.6d).

The Mastercard MP SDK provides MPA Developers with functionality to assist the MPA Developer in developing a Mastercard® Cloud-Based Payment (MCBP) solution as a mobile application for performing contactless transactions (Contactless Transactions) or a digital secure remote payment transactions (Remote Transactions (or (DSRP))) with a merchant using a mobile device.

To perform a Contactless Transaction, the solution developed by the MPA Developer will need to use Host Card Emulation (HCE) introduced in Android KitKat 4.4. More information on HCE can be found here: (https://developer.android.com/guide/topics/connectivity/nfc/hce.html).

The Mastercard MP SDK provides functionality which can assist the MPA Developer in building a solution with the following characteristics:

- Contactless and Remote Transactions in the same module
- Obtain payment credentials called Single Use Keys from the Cloud and use each of them for a single transaction
- Mobile Payment solution with Host Card Emulation
- Use NFC interface of Mobile Device
- Support PayPass, Mag Stripe and PayPass M/Chip (EMV)
- Online Authorized Payment Transactions (No Offline Transactions)
- Mobile PIN entry and Mobile CVM support simplifies the mobile payments value chain by reducing the number of participants
- Simple user installation/provisioning process
- User downloadable mobile application
- No TSM required
- Security mechanisms to mitigate the risk
- No change in acceptance environment

Architecture

This section describes the architecture of the MCBP modules and provides additional detail on each module. Figure 1 shows the software architecture of the Mastercard MP SDK and its Android modules as part of an example Android application with a UI (mcbp-sample-ui).
The Mastercard MP SDK is made up of the following modules.

- **Core SDK Modules/Projects**
  - `mobile-api-core` - General utility modules
  - `mobile-api-android` - Android specific implementation of the mobile-api-core components
  - `mcbp` - Core JAVA components including MPP, Mobile Kernel, LDE, and CMS <-> MPA communication protocol
  - `mcbp-android` - Android specific factories and modules. This also includes the C++ implementation of the MPP Lite
  - `mcbp-android-api` - SDK top level APIs to be used by 3rd party UI applications.

- **Helper Modules**
  - `mcbp-demo` - Utility APIs that are used to implement the UI demo mode. They allow to use selected SDK functionality without connecting to a CMS
  - `mcbp-demo-android` - Android specific implementation of the mcbp-demo components
  - `mcbp-wallet-services` - APIs related to wallet functionality. They are used to simulate certain Wallet aspects that are normally outside the scope of the SDK

- **Sample UI Application**
  - `mcbp-sample-ui` - UI sample Application. It provides examples on how to use MCBP components.
Mastercard MP SDK Functionality

The Mastercard MP SDK is capable of performing the following functionality:

1. Communicating with the CMS-D covering the following functions:
   a) Register
   b) Set Mobile PIN/Change Mobile PIN
   c) Request Session
   d) Provisioning of card profiles
   e) Notify Provision Result
   f) Replenish (fetching Transaction Credentials) - also includes providing status of Transaction Credentials provisioned.
   g) Delete card
   h) Get Task Status

2. Payment

   To facilitate the DSRP and contactless payment functionality using the provisioned card profile and credentials provided by the user.

   The developer of the MPA is then responsible for implementing the following:
   a) Communication with user from UI
   b) Communication with payment app server for registering wallet
   c) Generating a device fingerprint
   d) Security of user credentials
   e) Security of data obtained from the payment app server and the Mastercard MP SDK

Development Support

The Mastercard MP SDK for MCBP includes the following testing tools:
- Credential Management System (CMS) simulator
- Transaction Management System (TMS) Plugin (for Terminal Simulator)
- Sample game application to demonstrate Digital Secure Remote Payments (DSRP)

In addition, the following tools are also available to help with testing:
- Terminal Simulator, available at www.terminalsimulator.com
- Issuer Simulator, available at www.terminalsimulator.com

More information on the above tools can be found in [2].
Dependencies

The minimal configuration needed for Java Development Kit is given in [2], section “Dependencies”. The guidance provided in [2] and in this document assumes the MPA Developer’s system is already capable of building Android Applications using Android Studio.

The Mastercard MP SDK v1.0.6 supports minimum Android API Level 16 for DSRP payments and minimum Android API Level should be 19 for contactless payments.

Integration

The Mastercard MP SDK and the rest of the payment application developed by the MPA Developer will be merged by the MPA Developer and prepared for delivery to the user. For the source code building procedure the Mastercard MP SDK source code will be provided and its modules will be combined to be used in the chosen IDE (in this case Android Studio) following the procedure described in [2]. This means that the final developer that delivers the binary is responsible for its protection as listed also in Chapter 5 of this document MPA Developer Guidance.

Minimal Permissions Required

The following Android permissions are required by the Mastercard MP SDK from the user of the device:

```xml
<uses-permission android:name="android.permission.INTERNET" />
<uses-permission android:name="android.permission.NFC" />
```

The MPA requires also the following permission:

```xml
<uses-permission android:name="com.google.android.c2dm.permission.RECEIVE" />
```

Third Party Libraries

The Mastercard MP SDK relies on several 3rd party libraries. The following libraries are present as code within the Mastercard MP SDK code:

- Crypto++ 5.6.2
- Rapid XML

A list of the 3rd party libraries included in their original binary format (jar file) within the Mastercard MP SDK (distributed in compiled format as downloaded from public repositories) is given in the section “Third Party Licenses” of [2].
The list of assets to be protected by the Mobile Payment Application are as follows:

1. **Cryptographic Keys**: For each key the following attributes are identified: { Name of cryptographic key; Purpose; The key size and corresponding cryptographic algorithm when using the cryptographic key; The duration/longevity of the key; Where the cryptographic key is generated; and where it is stored and how }

<table>
<thead>
<tr>
<th>Key Name</th>
<th>Purpose</th>
<th>Key size / Algorithm</th>
<th>Key Duration</th>
<th>Where Generated</th>
<th>Stored, Encrypted</th>
<th>Used for Confidentiality, Integrity or Both</th>
</tr>
</thead>
<tbody>
<tr>
<td>Mobile Keys (Set Id)</td>
<td>Used to encrypt communication between CMS-D and MP SDK</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>Transport Key</td>
<td>Encryption of payload</td>
<td>128-bit AES</td>
<td>Long lived</td>
<td>CMS-D</td>
<td>LDE</td>
<td>Confidentiality</td>
</tr>
<tr>
<td>Data Encryption Key</td>
<td>Encryption of data fields</td>
<td>128-bit AES</td>
<td>Long lived</td>
<td>CMS-D</td>
<td>LDE</td>
<td>Confidentiality</td>
</tr>
<tr>
<td>MAC Key</td>
<td>Integrity and authenticity</td>
<td>128-bit AES</td>
<td>Long lived</td>
<td>CMS-D</td>
<td>LDE</td>
<td>Integrity</td>
</tr>
<tr>
<td>Mobile Session Keys</td>
<td>Used to respectively encrypt and decrypt any request payload sent to and received from MDES.</td>
<td>128-bit HmacSHA256</td>
<td>Short lived</td>
<td>MP SDK</td>
<td>Not Stored</td>
<td>Confidentiality</td>
</tr>
<tr>
<td>CMS-D Public Key</td>
<td>Used to encrypt RGK to the CMS-D</td>
<td>A certificate provided as a file of MIME type &quot;application/pkix-cert&quot; according to RFC 2585.</td>
<td>Long lived</td>
<td>N/A</td>
<td>Not Stored</td>
<td>Confidentiality</td>
</tr>
<tr>
<td>Random Generated Key (RGK)</td>
<td>Used to encrypt the Mobile keys during transport</td>
<td>128-bit AES</td>
<td>Short lived, erased after transporting Mobile keys to MPA</td>
<td>MP SDK</td>
<td>Not Stored</td>
<td>Confidentiality</td>
</tr>
<tr>
<td>Local Database Encryption Key (LDEK)</td>
<td>Used to encrypt the local database with the assets in it</td>
<td>256-bit AES</td>
<td>Long lived; for the lifetime of the MPA/wallet</td>
<td>MP SDK</td>
<td>Calculated at run-time using random generated</td>
<td>Both</td>
</tr>
<tr>
<td>Key Name</td>
<td>Purpose</td>
<td>Key size / Algorithm</td>
<td>Key Duration</td>
<td>Where Generated</td>
<td>Stored, Encrypted</td>
<td>Used for Confidentiality, Integrity or Both</td>
</tr>
<tr>
<td>-------------------------------------------------------</td>
<td>-------------------------------------------------------------------------</td>
<td>----------------------</td>
<td>-----------------------------</td>
<td>-----------------</td>
<td>------------------</td>
<td>------------------------------------------</td>
</tr>
<tr>
<td>ICC Key Encryption Key (IccKEK)</td>
<td>Used to encrypt/protect private components of the ICC key that is used by the payment application when signing data for CDA transactions</td>
<td>128-bit AES</td>
<td>Long lived; for lifetime of a card</td>
<td>CMS</td>
<td>LDE</td>
<td>Both</td>
</tr>
<tr>
<td>ICC Key components</td>
<td>Private components of the ICC key that is used by the payment application to create RSA key for signing data for CDA transactions</td>
<td>Size of each component depends on size of the RSA key. ICC Private Key CRT constants: - q⁻¹ mod p. - d mod (q – 1) - d mod (p – 1) - Prime factor q - Prime factor p</td>
<td>Long lived; for lifetime of a card</td>
<td>CMS</td>
<td>LDE</td>
<td>Confidentiality</td>
</tr>
<tr>
<td>Session Keys</td>
<td>Used for generation of one transaction cryptogram</td>
<td>TDES</td>
<td>Short-lived</td>
<td>MP SDK</td>
<td>Not stored</td>
<td>Confidentiality</td>
</tr>
<tr>
<td>Single Use Keys</td>
<td>Used in combination with the Mobile PIN to generate Session Keys</td>
<td>TDES</td>
<td>Short-lived</td>
<td>MP SDK</td>
<td>LDE</td>
<td>Confidentiality</td>
</tr>
</tbody>
</table>
2. **Session Code**: Received in RNS notification from CMS-D. Used to calculate the session key for communication with CMS-D by combining it with the MAC Key and Transport Key. Also used to derive the Authentication Code.

3. **Authentication Code**: Derived from session code, Mobile Key Set Id, and the device fingerprint. Used for securing HTTP communication with CMS-D.

4. **Card Profile**: Received in response to a card provision request and stored by the Mastercard MP SDK. Encrypted in transit using transport keys. Some parts of the profile data are in plaintext while other parts are either encrypted with a DEK key or an IccKEK key.

5. **Transaction Credentials**: Consists of Single Use Keys, ATC, IDN. These are received in a replenishment response, encrypted using the DEK key. Should be decrypted only during a transaction. Once decrypted, they need to be deleted from the system, even if not used.

6. **Mobile PIN**: Entered by user and passed to SDK for validation required for transaction processing. Needs to be cleared from memory immediately after use by Mastercard MP SDK to avoid potential leak of the value. Mobile PIN use: - SDK needs to know this to create Session Keys from Single Use Keys.

7. **Device Fingerprint**: Generated by the MPA and must be unique for each device.

8. **Mastercard MP SDK Source Code**: Shall be protected for integrity and confidentiality against reverse engineering attacks.

9. **Registration Code**: Used to activate and begin using a provisioned card to the MPA. Delivered in different out-of-band ways for example via e-mail or SMS.

10. **Protection Tools Parameters**: Parameters related to developer tools used for software protection of the MPA source code shall be properly protected. These are the parameters of software countermeasure tools that can be used to reverse engineer protections.
Chapter 4  Security Countermeasures

This chapter describes the security countermeasures that are used by the Mastercard MP SDK in order to reduce MPA and mobile device vulnerabilities. Some of these countermeasures are implemented by the Mastercard MP SDK while others are system specific and utilized by the Mastercard MP SDK.

Communication

Transport Layer
The Mastercard MP SDK uses the cryptographic providers present in the device to establish a TLS connection with the CMS-D to obtain payment credentials and store and handle them in a secure manner in the device. The communication with CMS-D uses SSL/TLS protocols with the option of preventing fallback to earlier versions. The Mastercard MP SDK will also handle certificate pinning however the input certificates must be provided by the MPA. The Mastercard MP SDK aims to verify that the domain name in the certificate received from the server matches the allowed host names specified by the MPA.

Application Layer
- **Key Exchange**: The Mastercard MP SDK is designed to generate a random key (RGK), which is encrypted using CMS-D public key and shared during the registration. The CMS-D in turn is designed to encrypt the mobile keys with the RGK before sharing with the Mobile Payment Application.
- **Dual Channel**: Assets like Registration Code, Session Code, and Mobile Keys are sent using Remote Notification Service.
- **Sessions**: Sessions are established using a Session Code and the validity is time bounded.
- **Communication**: The communication between CMS-D and Mastercard MP SDK is encrypted using the mobile keys. The data is encrypted using the transport key. The authenticity and integrity is verified using the MAC key. The sensitive data is further encrypted at the application data level by the data encryption key.
- **Ephemeral data**: The following data is only in volatile memory (RAM) temporarily:
  1. Random Generated Key (RGK)
  2. Session Code
  3. Registration Code
  4. Mobile PIN

  The lifetime of sensitive data is intended to be limited by clearing the memory containing them as soon as they are no longer required.

Storage Protection

The Mastercard MP SDK uses LDE Basic protection as described in [1] to encrypt its stored assets with the LDE Key. Assets are stored in an encrypted SQLite local database (LDE) in the storage domain of the MPA. The following data is encrypted by the LDE Key before it is stored in the encrypted database:

1. Mobile Keys
2. Card Profile
3. Transaction Credentials (including SUKs)

This database is encrypted by the LDE Key. The LDE Key is generated in a way bound to the device by its Device Fingerprint such that each device would generate a unique LDE Key. This means that the encrypted database of one device cannot be decrypted by another device. The AES LDE Key is generated using the following three elements:

1. Random generated data
2. Wallet developer input
3. ANDROID_ID

The Mobile PIN value entered through the device’s keyboard is combined with the SUK and intended to be encrypted immediately by a Session Key and sent to the host for verification with the Application Cryptogram. The MPA Developer may implement extra security countermeasures related to the capture of the Mobile PIN from the keyboard, such as a custom-built in-app keyboard.

Binary Protection

The MPA Developers receive the source code of the Mastercard MP SDK without any obfuscation applied to it.

Implementation of binary protection mechanisms is the responsibility of the MPA Developer which must integrate the source code of the Mastercard MP SDK and the MPA in the final binary deliverable. Once the MPA is integrated, the MPA Developer shall also be responsible for of the integrity protection of the binary during its lifecycle in the device.

Operating System Specific

The following functionalities of the Mastercard MP SDK may be realized by leveraging implementations available in the specific Android OS where the Mastercard MP SDK is running:

1. The Mastercard MP SDK uses the SQLite database implementation provided by the OS to create and update an encrypted database.
2. The ANDROID_ID is used to derive the unique key used to secure the database.
3. Mastercard MP SDK leverages the Android HTTP communication libraries for interaction with CMS-D.
4. The default HCE service within the Mastercard MP SDK leverages the HostApduService to communicate on the NFC interface.
Chapter 5  MPA Developer Guidance

Please note that the guidance below is not a replacement for the MPA Developer complying with the security guidelines set forth in the MCBP Security Guidelines for MPA Development [4] document in relation to the MPA, and is complementary to those guidelines.

The MPA Developer is solely responsible and liable for developing and implementing the security of the MPA and shall at all times ensure a level of security that: (i) complies with best industry practices and law; (ii) meets any specific threats to the MPA; (iii) adheres to the security guidance in this guide; and (iv) employs services, including authentication, integrity and confidentiality to safeguard against theft, misuse, unauthorized use and fraudulent and illegal activity. In an environment solely composed of software, it is common for developers to apply multiple layers of software security countermeasures, applied at an MPA level to increase the complexity for an attacker. It is the responsibility of the MPA developer to implement security countermeasures incorporating industry best practices.

Without limiting the foregoing paragraphs, the MPA Developer should implement the following security countermeasures in security sensitive areas of the code, for example, where plain text assets are being processed in order to comply with the MCBP Security Guidelines for MPA Development described in [4].

The MPA Developer is responsible for ensuring:

1. **Binary Protections:** Implementation of secure binary protection mechanisms in order to prevent code reverse engineering which will integrate the source code of the Mastercard MP SDK and the MPA in the deliverable binary. Once the MPA is integrated, the MPA Developer shall also be responsible for the integrity protection of the binary during its lifecycle in the device. This includes static and runtime protection of the application and its resources.

2. **Protect sensitive code and functionality:** The MPA application provides appropriate protection for the CVM mechanism and Mastercard MP SDK APIs, for example by use of obfuscation.

3. **Local Database Encryption:** The Mastercard MP SDK implements storage protection using LDE Basic as defined in [1]. However to reach a satisfactory level of security meeting the Security Guidelines for MPA Development in [4], LDE Advanced should be implemented. LDE Advanced uses advanced mobile application software development techniques for example White-Box Cryptography (WBC) as a solution leveraging support of dynamic key integration to deliver a diversified LDE key [1].

4. **Application Integrity Protection:** The source code and resources of the MPA do not undergo unauthorized modification or update during the application lifecycle. Integrity checks may consist of cryptographic verification at run-time of the MPA.

5. **Cryptographic Provider:** The Mastercard MP SDK utilizes the default cryptographic provider in the Android OS to perform all the cryptographic operations. The MPA Developer may implement (use) and maintain another cryptographic provider as part of the MPA in order to perform functions related to encryption, (H)MAC, hashing algorithms etc. and cipher-suites used for the transport layer security.

6. **Certificate Pinning:** The certificate pinning mechanism is implemented in the Mastercard MP SDK. The MPA Developer shall provide the proper certificates for pinning.
7. **Root Kit Detection**: Implementation device secure root kit detection. This may consist of simple detection of a rooted environment or may include other parameters in a complete device attestation.

8. **Debug Detection**: Implementation of a secure debug detection. The MPA should include code such that if a debugger is attached then it should stop the execution.

9. **Hooking detection**: Implementation of a secure hook detection. This may be applied to specific security related functions in the Mastercard MP SDK such as cryptographic functions or sensitive data reading/writing (in a file or database). A limitation is that the detection may affect the performance of the function which it protects.

10. **Android Permissions**: That it does not request the user to grant unnecessary Android permissions.

11. **Coding Guidelines**: That the following secure coding practices are followed by the MPA Developers:

    1. Sensitive information must not be logged
    2. Exceptions must not expose sensitive information
    3. Do not use strings to store sensitive data; strings are immutable and cannot be cleared
    4. Minimize the accessibility of classes and their members.

12. **Device Fingerprint**: That there is a unique and confidential device identifier used by the MPA in an individualization technique to prevent its execution on an identical device. The Device Fingerprint is generated by a deterministic function and used as a diversifier in the generation of the LDE encryption key. Since the Fingerprint cannot be stored in the LDE, the MPA Developer must avoid storing the Fingerprint for future use in unprotected storage in the device.

13. **Key Logging**: Implementation of extra security countermeasures related to the capture of the Mobile PIN from the keyboard as required, such as a custom-built in-app keyboard.

**Additional Security Measures**

It is the responsibility of the MPA Developer to implement the security countermeasures incorporating industry best practices. The following is a non-exhaustive list of security countermeasures:

1. **Clipboard attacks**: Protection from device clipboard attacks by disabling copy/paste functionalities in the context of the MPA. If at any point of the MPA usage an asset needs (for practical reasons) to be copied/pasted, protection may be necessary.

2. **Screen Overlay attacks**: Protection from screen software overlay attacks (for example Acecard) if necessary.

3. **Screenshots of sensitive data**: Protection of screen from system-wide readable screenshot attacks. This is related to the in-app screenshots that Android OS takes in the application the moment it is put in the background. If at any point of the MPA usage an asset is shown on the screen this protection may be necessary.
4. **Android Keystore**: The sensitive data can be encrypted by keys generated in the Android Keystore [5]. The Android Keystore can be used in combination with the LDE as an additional security measure. Please note that Android Keystore has not been evaluated for security by Mastercard.

5. **White-Box Cryptography**: White-Box cryptography can be used to protect the cryptographic keys that are stored in the device by the MPA. Limitations are that it could result in increased application size and it may affect performance.