EMV® Specification Bulletin No. 280

August 2023

EMV® 3-D Secure SDK Specification version 2.3.1.1

This Specification Bulletin No. 280 provides the updates, clarifications and errata incorporated into the EMV® 3-D Secure SDK Specification since version 2.2.0

Applicability

This Specification Bulletin applies to:

- *EMV® 3-D Secure SDK Specification, Version 2.2.0*
- *EMV® 3-D Secure SDK Specification, Version 2.3.0.0*
- *EMV® 3-D Secure SDK Specification, Version 2.3.1.0*
- *EMV® 3-D Secure SDK Specification, Version 2.3.1.1*

Updates are provided in the order in which they appear in the specification. Deleted text is identified using strikethrough, and red font is used to identify changed text. Unedited text is provided only for context.

Related Documents

EMV® 3-D Secure SDK Specifications, Versions 2.2.0, 2.3.0.0, 2.3.1.0 and 2.3.1.1

Effective Date

- *August 2023*

Contents

Throughout Specification

- Revisions added to improve grammar, spelling, consistency, clarity and readability without any effect on the meaning or interpretation of the specification are not included in this specification bulletin.
- References to Sections and Tables in the *EMV® 3-D Secure Protocol and Core Functions* have been updated as applicable.
- For consistency and ease of reference, the *EMV® 3-D Secure Protocol and Core Functions Specification* is referred to as the *Core Specification*.
- Instances of “Mobile” SDK were updated to “Default” SDK.
- Code samples in Annex C to the Specification have been updated as applicable and are not replicated in this specification bulletin.

Chapter 1 Introduction

1 Introduction

~~The past few years have seen a dramatic rise in the use of mobile devices. A growing number of consumers now purchase products and log in to numerous online services through mobile apps. There is a need to improve authentication security in mobile-based apps.~~

1.1 Purpose

~~This document describes the specification for the 3DS SDK. Enhancements to the *EMV 3-D Secure Protocol and Core Functions Specification version 2.2.0* (later referred to as *EMV 3DS Protocol Specification* in this document) that have an impact on the SDK will be included in later versions of this document.~~

This document specifies the requirements for the EMV 3-D Secure Default SDK.

For purposes of this document, when the phrase 3-D Secure, and/or 3DS is utilised, the intent is EMV 3-D Secure.

1.3 Normative References

~~The following standards contain provisions that are referenced in this specification. The latest version including all published amendments shall apply unless a publication date is explicitly stated.~~

For the standards containing provisions that are referenced in this specification refer to Table 1.1 of the *EMV® 3-D Secure Protocol and Core Functions Specification* (hereinafter referred to as the *Core Specification*).

Table 1.1 Normative References was deleted.

1.5 Abbreviations

~~The abbreviations listed in Table 1.2 are used in this specification.~~

For the abbreviations used in this specification, refer to Table 1.4 of the *Core Specification*.

Table 1.2 Abbreviations was deleted.

1.6 Supporting Documentation

The following documents are specific to the EMV 3-D Secure protocol and should be used in conjunction with this specification. These documents as well as *the EMV® 3-D Secure Frequently Asked Questions* are located on the EMVCo website under the 3-D Secure heading.

- *EMV® 3-D Secure—Protocol and Core Functions Specification*
- *EMV® 3-D Secure SDK Technical Guide*
- *EMV® 3-D Secure SDK—Device Information*
- *EMV® 3-D Secure Split-SDK Specification*
- *EMV® 3-D Secure JSON Message Samples*

1.7 Terminology and Conventions

3DS SDK

When this specification refers to the 3DS SDK, EMVCo has defined two options for a 3DS SDK implementation. The options are as follows:

1. **Default-SDK**—Software component designed as an SDK that is integrated into a 3DS Requestor App. In earlier versions of the *Core Specification*, this is referred to as the 3DS SDK.
2. **Split-SDK**—Client-server implementation of the 3DS SDK. Some functions of the Split-SDK entity can be performed by either the Split-SDK Client or the 3DS Split-SDK Server or, in some situations, both. This SDK option is defined in the *EMV 3-D Secure—Split-SDK Specification*.

Unless explicitly noted otherwise, the term 3DS SDK applies as identified above.

Refer to the applicable SDK specification for detailed information regarding either SDK option.

Activate the 3DS SDK

Detailed information about the 3DS SDK activation can be obtained in the applicable 3DS SDK specification. When the *Core Specification* refers to "Activate the 3DS SDK" in the context of a SDK implementation, the applicable action is the `initialize` method as defined in Section 4.1.1 of this specification.

Perform the Challenge

Detailed information about the 3DS SDK performing the challenge can be obtained in the applicable 3DS SDK specification. When the *Core Specification* refers to "Perform the Challenge" in the context of an SDK implementation, the applicable action is the `doChallenge` method as defined in Section 4.4.2 of this specification.

Chapter 2 EMV 3-D Secure Architecture and Flows Overview

2.3 UI Types for Challenge Flow

2.3.2 Challenge Flow Implemented Using HTML UI

The last sentence in the section was revised to refer to a different section in the Core Specification.

For more information about the HTML UI, refer to Section ~~4.2.4~~4.2.5 of the *Core Specification*.

Chapter 3 Getting Started with the EMV 3-D Secure ~~Mobile~~ Default SDK

Table 1.1 3DS SDK Lifecycles was deleted in its entirety.

3.3 Authentication Flows

3.3.1 Frictionless Flow

13. In the `getProgressView` method call, the 3DS SDK ~~shall~~ returns an instance of Progress View (processing screen). Refer to ***[Req 141]*–*[Req 144]*** in the *Core Specification* for additional detail.

~~***[Req 6]***~~ Return an instance of Progress View (processing screen). ~~The progress view shows the Cardholder that an activity is being processed. The 3DS SDK shall create the Progress View object and return a handle of this object to the app.~~

3.3.2 Challenge Flow

21. The 3DS SDK ~~shall **[Req 8]**~~ initiates the Challenge Flow by displaying the UI for the challenge screens as defined in ***[Req 358]*** of the *Core Specification*.

22. The Cardholder responds to the challenge.

23. The 3DS SDK ~~shall **[Req 9]**~~ uses a graphical element (a processing view) on the Challenge screen to show that the Cardholder's response is being processed. Refer to ***[Req 147]***, ***[Req 148]*** and ***[Req 151]*** in the *Core Specification* for additional detail.

3.4 Summary of 3DS SDK Code Elements

3.4.4 Enum Summary

Table 3.4: Enum

Requirement ID	Enum	Description
[Req 34]	`UICustomizationType`	This enum shall define the UICustomization type. For detailed information, see `Enum UICustomizationType`.

Chapter 4 Code Elements of the EMV 3-D Secure Default SDK

4.1 Interface ThreeDS2Service

4.1.1 initialize

The following Android code snippet shows the signature of the `initialize` method:

```
public void initialize(android.content.Context
~~a~~ApplicationContext, ConfigParameters configParameters, String
locale, Map<UICustomizationType, UiCustomization>
uiCustomizationMap) throws InvalidInputException,
SDKAlreadyInitializedException, SDKRuntimeException
```

Table 4.2: initialize Parameters

Parameter	Mandatory	Description
`uiCustomizationMap`	No	UI configuration information that is used to specify the UI layout and theme for a UICustomization Type. For example, font style and font size for DEFAULT or DARK mode. For more information, see `Class UiCustomization`, `Enum UICustomizationType`.

4.2 Class ConfigParameters

Table 4.10: ConfigParameters Class Methods

Method	Description
`getParamValue`	Returns a configuration parameter's value either from the specified group or from the default group.

4.4 Interface Transaction

4.4.3 getProgressView

The `getProgressView` method ~~shall return~~returns an instance of Progress View (processing screen) that the 3DS Requestor App uses. The processing screen displays the ~~Directory Server~~Payment System logo, and a graphical element to indicate that an activity is being processed.

4.5 Class UiCustomization

The following Java code snippet shows the definition of the `UiCustomization` class:

```
public class UiCustomization {


    public enum ButtonType {SUBMIT, CONTINUE, NEXT, CANCEL, 
RESEND}, OPEN_OOB_APP, ADD_CH}
    public enum UICustomizationType {DEFAULT, DARK, MONOCHROME}
    public void setButtonCustomization(…)
    public void setToolbarCustomization(…)
    public void setLabelCustomization(…)
    public void setTextBoxCustomization(…)
    public ButtonCustomization getButtonCustomization()
    public ToolbarCustomization getToolbarCustomization()
    public LabelCustomization getLabelCustomization()
    public TextBoxCustomization getTextBoxCustomization()
    }
```

4.5.2 setButtonCustomization—variation

4.8 Class ToolbarCustomization

4.8.4 getHeaderText

getHeaderText Return Value

The `getHeaderText` method returns the header text (as a String) of the toolbar. If the header text is blank or null, the default value of "SECURE CHECKOUT" or its localised equivalent is used by the 3DS SDK.

4.23 Enum ButtonType

Table 4.103: ButtonType Enum

Button Type	Description
`OPEN_OOB_APP`	Open OOB App button
`ADD_CHOICE`	Additional Choice button

4.24 Enum UICustomization Type

The `UICustomizationType` enum defines the UICustomization types. These types indicate when the `UiCustomization` object is applicable, depending on the SDK UI mode.

```
The following Java code snippet shows the definition of the
UICustomizationType enum:
public enum UICustomizationType {DEFAULT, DARK, MONOCHROME}
```

Table 4.103 summarises the UICustomization types that are defined by the `UICustomizationType` enum.

Table 4.104: UICustomizationType Enum

UICustomization Type	Description
`DEFAULT`	Customization for SDK UI in default mode
`DARK`	Customization for SDK UI in dark mode
`MONOCHROME`	Customization for SDK UI in monochrome mode

Chapter 5 Message Processing

~~*[Req 70]* During Out-of-Band (OOB) authentication, when the 3DS Requestor App comes to the foreground, the CReq shall automatically be submitted to the ACS and the value of the OOB Continuation Indicator field shall be set to true.~~

For the processing of the OOB challenge, refer to the *Core Specification*.

Chapter 7 User Interface

~~**[Req 34]**~~

~~The 3DS SDK shall render the UI for the Challenge Flow in one of the following formats:~~

- ~~• HTML UI, in which the Cardholder challenge is applied by using an HTML-based user interface.~~
- ~~• Native UI, in which the Cardholder challenge is applied by using a native user interface.~~

~~The UI format to be displayed by the 3DS SDK is determined based on the ACS UI Type value obtained as part of the CRes message. For information about the ACS UI Type, see the ACS UI Type row in Table A.1, "EMV 3-D Secure Data Elements" in the EMV 3DS Protocol Specification.~~

The 3DS SDK renders the UI for the Challenge Flow. The UI format to be displayed by the 3DS SDK is determined based on the ACS UI Type value obtained as part of the CRes message. For information about the ACS UI Type, see Table A.1 in the *Core Specification*.

Note: The implementer~~ shall~~ must consider regional Accessibility rules while developing the user interface.

~~**Note: All Device Rendering Options supported shall be supported by the SDK. This is also mentioned in [Req 314] in the *EMV 3DS Protocol Specification*.**~~

The 3DS SDK must support the UI element types listed in Table A.19 in the *Core Specification*.

~~**[Req 35]**~~

~~The 3DS SDK shall support the UI element types listed in Table 7.1.~~

Table 7.1: UI Element Types was deleted.

7.1 HTML UI

~~**[Req 36]**~~

~~The HTML UI shall be rendered in a web view controlled by the 3DS SDK.~~

~~During the Challenge Flow, the ACS provides the content that is displayed to the Cardholder. This content is provided as a fully formed HTML snippet. The ACS encrypts the HTML snippet and transmits it to the 3DS SDK in the CRes message. The 3DS SDK decrypts the HTML snippet and use a web view to display the content on the mobile device. The 3DS SDK shall display the HTML exactly as provided by the Issuer.~~

~~The **ACS HTML** field in the CRes message holds the HTML snippet to be displayed to the Cardholder.~~

~~**[Req 37]**~~

~~The Cardholder data (response) shall be captured and sent to the ACS in the CReq message. The SDK shall not modify this response data before passing it in the **Challenge HTML Data Entry** field of the CReq message. This field holds the Cardholder's challenge response.~~

~~When the Cardholder's response is returned as a parameter string, the form data is passed to the web view instance by triggering a location change to a specified (HTTPS://EMV3DS/challenge) URL with the challenge responses appended to the location URL as query parameters (for example, HTTPS://EMV3DS/challenge?city=Pittsburgh). The web view instance, because it monitors URL changes, receives the Cardholder's responses as query parameters.~~

~~**[Req 38]**~~

~~The header for the HTML UI pages that are rendered by the 3DS SDK shall not occupy more than 10% of the screen height.~~

The HTML UI is rendered in a web view controlled by the 3DS SDK. Refer to Section 4.2.5 and ***[Req 371]*** in the *Core Specification* for additional detail.

The Cardholder data (response) is captured and sent to the ACS in the CReq message. For information about the steps following the Cardholder's submission of their response, refer to Section 4.2.7.1 and ***[Req 171]*** of the *Core Specification*.

7.2 Native UI

~~***[Req 39]***~~ The Native UI ~~shall be~~ is rendered and controlled by the 3DS SDK.

The Native UI integrates into the 3DS Requestor App UI to facilitate a consistent user experience. ~~The Native UI has a similar look and feel as the 3DS Requestor's App with the authentication content provided by the Issuer.~~

~~This format also allows for Issuer and Payment System branding. The 3DS SDK controls the rendering of the UI such that the authentication pages inherit the 3DS Requestor's UI design elements.~~ The CRes message carries the information that is required to render the UI. For more information about Native UI, refer to Section 4.2 of the *Core Specification*.

~~The **Challenge Selection Information** field in the CRes message holds the selection information that is presented to the Cardholder if the challenge type is single select or multi select. UI text, such as label names, questions, and help text, is sent in a JSON array. The 3DS SDK parses the UI text and then displays it in the user interface.~~

~~***[Req 40]*** The Cardholder data (response) shall be captured and sent to the ACS in the CReq message. The **Challenge Data Entry** field in the CReq message holds the Cardholder's challenge response. If the cardholder has submitted the response without entering any data in the UI, the **Challenge Data Entry** field shall not be present in the CReq message.~~

~~***[Req 71]*** *If the Cardholder does not enter any data in the UI, the **Challenge No Entry** field shall be sent in the CReq message with the value "Y".*~~

Native UI Customisation

[Req 42] The 3DS SDK shall allow customization of the following UI elements on the challenge screens. The information required for UI customization is passed to the 3DS SDK during initialisation.

- Text font (for Label Text, Button Text, Textbox Text)
- Text size (for Label Text, Button Text, Textbox Text)
- Text colour (for Label Text, Button Text, Textbox Text)
- Button Style
- Textbox Style
- Toolbar

For more information about UI elements customization, see Section 4.5.

7.2.1 Input and Output Formats for Native UI

Single Text Input

In this example, the following are the field formats:

Challenge Data Entry or Challenge Data Entry 2 field:

7.2.2 UI Templates for Native UI

~~***[Req 41]***~~ The 3DS SDK ~~shall~~must have predefined UI templates for the challenge screens for each challenge type. For more information about Native UI templates, refer to Section 4.2.2 of the *Core Specification*. ~~Based on the Issuer's choice of template for each challenge type (as determined by the ACS UI type element in the CRes message), the 3DS SDK shall render the UI for the challenge screens. UI elements should be placed in a logical order in the templates. The placement should conform to the standard UI best practices of the country or region where the 3DS SDK will be used.~~

~~The 3DS SDK should fine-tune the rendering of the UI on the Cardholder device. The 3DS SDK can optimise the content provided by the Issuer (for example, by removing an extra line feed that would cause scrolling). The formatting provided in the CRes message need not be exactly what is displayed to the Cardholder.~~

~~For more information about Native UI templates, refer to Section 4.2.2, "Native UI Templates" in the EMV 3DS Protocol Specification.~~

~~**Note: The use of a carriage return in any UI data element is permitted only as specified in Table A.1, "EMV 3-D Secure Data Elements" in the EMV 3DS Protocol Specification.**~~

~~7.3 UI Elements Customization~~

~~***[Req 42]*** The 3DS SDK shall allow customization of the following UI elements on the challenge screens. The information required for UI customization is passed to the 3DS SDK during initialization.~~

- ~~Text font (for Label Text, Button Text, Textbox Text)~~
- ~~Text size (for Label Text, Button Text, Textbox Text)~~
- ~~Text colour (for Label Text, Button Text, Textbox Text)~~
- ~~Button Style~~
- ~~Textbox Style~~
- ~~Toolbar~~

~~*Annex C EMVCo Testing and Approval*~~

This Annex was entirely removed. Annex D was then renamed to Annex C.

Legal Notice