EMV® Specification Bulletin No. 279

August 2023

EMV® 3-D Secure Protocol and Core Functions Specification version 2.3.1.1

This Specification Bulletin No. 279 provides the updates, clarifications and errata incorporated into the EMV® 3-D Secure Protocol and Core Functions Specification since version 2.2.0 (as amended by Specification Bulletin No. 214v3)

Applicability

This Specification Bulletin applies to:

- *EMV® 3-D Secure Protocol and Core Functions Specification, Version 2.2.0*
- *EMV® 3-D Secure Protocol and Core Functions Specification, Version 2.3.0.0*
- *EMV® 3-D Secure Protocol and Core Functions Specification, Version 2.3.1.0*
- *EMV® 3-D Secure Protocol and Core Functions 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. Green double underline is used to indicate moved text. Unedited text is provided only for context.

Related Documents

EMV® 3-D Secure Protocol and Core Functions 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

- To facilitate enhanced version number management, a fourth digit was added to the 3-D Secure Protocol and Core Functions Specification version number: 2.3.1.x.
- Data element name/terminology updates:
  - ACS Start Protocol Version/ACS End Protocol Version data elements were updated to a single element: ACS Protocol Version
  - DS Start Protocol Version/DS End Protocol Version were updated to a single element: DS Protocol Version
  - BIN range was changed to card range.
  - All instances of *White List*, *whitelisted*, *whitelisting* were updated to *Trust List*. Figure 4.26 and Figure 4.27 were updated to include this data element update.
  - All instances of challenge window were changed to challenge iframe.
  - Annex A Section and Table references may be updated throughout the specification to reflect changes made in Annex A for version 2.3.1.0.
  - Instances of SDK were replaced with 3DS SDK.
- In Section 5.9, references to Section 5.1.6 were replaced with references to Section 5.1.7.
- Revisions added to improve grammar, consistency, clarity and readability without any effect on the meaning or interpretation of the specification are not included in this bulletin.
- Updates made to defined abbreviations, such as EC(C) and DH (Diffie–Hellman), have no substantive effect on the use of the underlying specification and are not reflected in this bulletin.

Chapter 1 Introduction

The 3-D Secure authentication protocol can be initiated through three Device Channels:

- **Browser-based**—Authentication during a transaction on a Consumer Device that originates from a website utilising a ~~browser~~ Browser as defined in Table 1.3.

1.3 Normative References

Table 1.1 Normative References

Reference	Publication Name	Bookmark
IETF BCP 47	*Tags for Identifying Languages*	https://tools.ietf.org/html/bcp47
RFC 2397	*The "data" URL scheme*	https://datatracker.ietf.org/doc/html/rfc2397
RFC 3986	*Uniform Resource Identifier (URI): Generic Syntax*	https://tools.ietf.org/html/rfc3986
RFC 791	*INTERNET PROTOCOL*	https://tools.ietf.org/html/rfc791
RFC 4291	*IP Version 6 Addressing Architecture*	https://tools.ietf.org/html/rfc4291
RFC 7233	*Hypertext Transfer Protocol (HTTP/1.1): Range Requests*	https://datatracker.ietf.org/doc/html/rfc7233

1.4 Acknowledgements

The following ISO Standards are referenced in this specification. The latest version including all published amendments shall apply unless a publication date is explicitly stated.

Table 1.2 ISO Standards

Reference	Publication Name	Bookmark
ISO/IEC 7812-1~~:2015~~	*ISO/IEC 7812-1~~:2015~~ Identification cards—Identification of issuers—Part 1: Numbering system*	
ISO/IEC 7813~~:2016~~	ISO/IEC 7813~~:2016~~ *Information technology—Identification cards—Financial transaction cards*	
ISO/IEC 7816-5~~:2004~~	ISO/IEC 7816-5~~:2004~~ *Identification cards—Integrated circuit cards—Part 5: Registration of application providers*	
ISO 8583-1	ISO 8583-1 *Financial transaction card originated messages — Interchange message specifications — Part 1: Messages, data elements and code values*	https://www.iso.org/standard/31628.html

1.5 Definitions

Table 1.3 Definitions

Term	Definition
3DS SDK	~~3-D Secure Software Development Kit (SDK).~~ A component that ~~is incorporated into~~interacts with the 3DS Requestor App. The 3DS SDK performs functions related to 3-D Secure on behalf of the 3DS Server.
Access Control Server User Interface (ACS UI)	The ACS UI is generated during a Cardholder challenge and is rendered by the ACS within a Browser challenge ~~window~~iframe.
App Screen Orientation	The orientation of the app screen display on the device, which may differ from the device orientation (for example, if the app supports Portrait-only or Landscape-only display, or if the device is in multi-window or split-screen mode). The orientation is considered Landscape if the display is wider than it is tall, and Portrait otherwise.
Bank Identification Number (BIN)	The first six or eight digits of a payment card account number that uniquely identifies the issuing financial institution.
Base64url	Encoding applied to the 3DS Method Data, Device Information, WebAuthn Credential List and the CReq/CRes messages as defined in RFC 7515.
Card Range Data File	The file containing the JSON Card Range Data object. The Card Range Data provides to the 3DS Server the 3DS protocol versions supported by the card ranges hosted by the ACS, and other optional information (e.g. 3DS Method, Message Extension).

Term	Definition
Decoupled Authentication Fallback	An additional challenge option for an ACS during the Challenge process. By returning Transaction Status = D in the RReq message, the ACS requests that the 3DS Server initiate a subsequent 3DS authentication using Decoupled Authentication.
Device Binding	In this specification, the process to link the Consumer Device used for a transaction to the Cardholder Account and/or Cardholder.
Ends processing	In the 3-D Secure processing flow, this indicates that an error has been found by a specific 3-D Secure component, which reports the error via the appropriate Error Message as defined in Section ~~A.5.5~~A.9 or RReq message as defined in Table B.8.
Fully Qualified URL	A Fully Qualified URL contains all the information necessary to locate a web resource ~~using the following format: scheme://server/path/resource,~~ and is defined as an 'Absolute-URL string' with scheme 'https', encoded in 'UTF-8' using 'url-code-points' from https://whatwg.org/. Refer to https://url.spec.whatwg.org/#absolute-url-string and to https://url.spec.whatwg.org/#url-code-points A Fully Qualified URL does not contain credentials (https://url.spec.whatwg.org/#include-credentials). Example: ~~https://server.domainname.com/acs/auth.html~~https://server.domainname.com/acs/auth%20(*ret
iframe	An iframe (short for inline frame) is a frame within a frame. It is used to embed a piece of HTML content from other sources in an HTML document. Refer to: w3c: https://www.w3.org/html/wg/spec/the-iframe-element.html#the-iframe-element OR whatwg: https://html.spec.whatwg.org/#the-iframe-element
OOB Authentication App	App on a Consumer Device that is used by the ACS to authenticate the Cardholder as part of the 3-D Secure flow, for example, a mobile banking app. See Section 3.2 for details of the OOB flow.
Operation Request (OReq) Message	The OReq message sequence is created to communicate operational information serving as an alert, a reminder, report, or call to action. This message is not part of the 3-D Secure authentication message flow.
Operation Response (ORes) Message	The ORes message acknowledges receipt of the OReq message sequence. The message is created by the recipient of the OReq message and sent to the source of the OReq message.
Platform Provider	An entity that provides a digital ecosystem consisting of an operating system and/or hardware components, capable of uniquely identifying the consumer and their device through a user ID and a hardware-derived device ID, and sharing these IDs for the purposes of risk assessment and fraud prevention.

Term	Definition
Preparation Response (PRes) Message	Response to the PReq message that contains the DS Card Ranges, active Protocol Versions for the ACS and DS and 3DS Method URL, or a Card Range Data File URL to download this information, so that updates can be made to the 3DS Server's internal storage.
Protocol Version	~~Refers to the version of the EMV 3-D Secure specification that the component supports. The protocol version for this specification is 2.1.0.~~ Defines the message interoperability between the EMV 3-D Secure components.
Responsive Design	Responsive design is an approach to make the web page content adjust to the dimensions of the device's screen for a better user experience. The approach is based on the use of three web techniques when designing the web pages: • Flexible grid to create the web page layout that dynamically adapt to the screen width. • Media queries to allow the page to adopt different CSS styles depending on the Browser and device screen. • Flexible media to make images scalable to the size of the viewport.
Secure Payment Confirmation	FIDO-based authentication to securely confirm payments initiated via the Payment Request API on a Browser (refer to w3.org for additional information).
Token Service Provider	A role within the Payment Tokenisation ecosystem that is authorised by a Token Programme to provide Payment Tokens to registered Token Requestors. Refer to the *EMV® Payment Tokenisation Specification - Technical Framework*.
Trust List ~~Whitelisting~~	In this specification, the process of ~~an ACS~~ enabling the Cardholder to place the 3DS Requestor on their trusted beneficiaries list.
WebAuthn	Defines an API enabling the creation and use of strong, attested, scoped, public key-based credentials by web applications, for the purpose of strongly authenticating users. Refer to https://www.w3.org/TR/webauthn-2/

1.6 Abbreviations

Table 1.4 Abbreviations

Abbreviation	Description
AOC	Attestation of Compliance
~~CA-DS~~	~~Certificate Authority Directory Server~~
CEK	Content Encryption Key
DH	Diffie–Hellman

Abbreviation	Description
DS CA	Directory Server Certificate Authority
ECC	Elliptic Curve Cryptography
LOA	Letter of Approval
OReq	Operation Request Message
ORes	Operation Response Message
SPC	Secure Payment Confirmation

1.7 3-D Secure Protocol Version Number

~~The following table provides the Protocol Version Number status for the EMV 3-D Secure Protocol and Core Functions Specification.~~ Refer to *EMV® Specification Bulletin 255* for the list of active Protocol Version Numbers.

Table 1.5 Protocol Version Numbers was removed from the specification.

1.8 Supporting Documentation

- *EMV® 3-D Secure—Split-SDK Specification*
- *EMV® 3-D Secure Message Extensions*
  - *EMV® 3-D Secure Bridging Message Extension*
  - *EMV® 3-D Secure Device Acknowledgement Message Extension*
  - *EMV® 3-D Secure Payment Token Message Extension*
  - *EMV® 3-D Secure Travel Industry Message Extension*
- *EMV® Specification Bulletin 255—3-D Secure Protocol Version Numbers*

1.9 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. This SDK option is defined in the *EMV 3-D Secure—SDK Specification*, in which it is referred to as the 3DS SDK. In earlier versions of this *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 a Split-SDK Client or a Split-SDK Server or, in some situations, both. The Split-SDK has multiple variants depending on the Consumer Device and the 3DS Requestor Environment. These variants include the Split-SDK/Native, Split-SDK/Shell and Split-SDK/Browser, and each 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 3DS SDK specification for detailed information regarding the SDK options.

Activate(s) the 3DS SDK

Detailed information about the 3DS SDK activation can be obtained in the applicable 3DS SDK specification.

Perform(s) the Challenge

Detailed information about the 3DS SDK performing the challenge can be obtained in the applicable 3DS SDK specification.

1.10 Constraints

The *Core Specification* or any implementation of the *Core Specification* is not intended to replace or interfere with any international, regional, national or local laws and regulations; those governing requirements supersede any industry standards.

Chapter 2 EMV 3-D Secure Overview

2.1 Acquirer Domain

2.1.1 3DS Requestor Environment

2.1.1.1 3DS Requestor

To process 3-D Secure transactions:

- **App-based**—3DS Requestor App integrates with the 3DS SDK as defined in the applicable ~~EMV 3-D Secure~~ 3DS SDK ~~Specification~~specification. The 3DS SDK displays the User Interface (UI) to Cardholders.

2.1.2 3DS Integrator (3DS Server and 3DS Client)

The 3DS Integrator provides the approved 3DS SDK component or the 3DS Method functionality to 3DS Requestors for integration ~~into~~ with their 3DS Requestor App and/or website.

2.2 Interoperability Domain

2.2.1 Directory Server

The DS performs a number of functions that include:

- Maintaining ACS and DS ~~Start and End~~ Protocol Version lists and 3DS Method URLs

2.2.2 Directory Server Certificate—Authority

These certificates include:

- TLS client and server certificates used in the communication between the 3DS Server and the DS, and between the DS and the ACS.
- ~~Signing~~ Certificates used to sign ~~messages~~ data elements passed from the ACS to the 3DS SDK.
- Certificates used to sign data elements passed from the 3DS SDK to the DS.

2.4 3-D Secure Messages

2.4.1 Authentication Request Message (AReq)

There is only one AReq message per authentication, except for the 3DS Requestor-Initiated SPC Authentication and Decoupled Authentication Fallback.

2.4.2 Authentication Response Message (ARes)

There is only one ARes message per ~~transaction~~authentication, except for the 3DS Requestor-Initiated SPC Authentication and Decoupled Authentication Fallback.

2.4.3 Challenge Request Message (CReq)

- **Browser-based** – The CReq message is ~~sent~~ formed by the 3DS Server and is posted through the Cardholder Browser. There is only one CReq message per challenge.

2.4.5 Results Request Message (RReq)

There is only one RReq message per ~~AReq message~~3DS transaction.

2.4.9 Operation Request Message (OReq)

The OReq message sequence is created to communicate operational information serving as an alert, a reminder, report, or call to action. This message is not part of the 3-D Secure authentication message flow.

2.4.10 Operation Response Message (ORes)

The ORes message acknowledges receipt of the OReq message sequence. The message is created by the recipient of the OReq message sequence and sent to the source of the OReq message.

2.6 Frictionless Flow Outline

2.6.2 3DS Requestor Environment—Browser-based

1.1 **3DS Requestor and 3DS Server**—The 3DS Requestor communicates with the 3DS Server. The 3DS Server determines the ACS and DS ~~Start and End~~ Protocol Version(s) and, if present, obtains the 3DS Method URL for the requested card range and returns the information to the 3DS Requestor.

2.7 Challenge Flow Outline

New Note was added after Step 6:

Note: For the App-based model, Step 5 and Step 6 will be repeated until the ACS makes a determination.

Note: For the Browser-based model, the CRes message is sent after Step 8.

Chapter 3 EMV 3-D Secure Authentication Flow Requirements

For an App-based model, also refer to the applicable ~~EMV 3-D Secure~~ 3DS SDK ~~Specification~~ specification for detailed requirements and implementation guidelines.

3.1 App-based Requirements

Step 2: The 3DS Requestor App

~~The 3DS Requestor App uses the Cardholder Account Number and optionally other cardholder information to identify the Payment System. Payment Systems are identified by their ISO RID (as defined in Table 1.2).~~

The 3DS Server provides to the 3DS Requestor App through the 3DS Requestor Environment:

- the Directory Server ID value (which is the Payment System's RID) that is used to identify the key for the Device Information encryption.
- the Message Version Number used in the CReq/CRes messages.

The 3DS Server shall:

Requirement 11 was moved from Step 5 to Step 2. New Requirement 419 was added directly after Requirement 11.

[Req 11]

Determine which DS the authentication transaction needs to be sent based on the BIN (as defined in ISO 7812) and optionally other Cardholder Account Information.

[Req 419]

Use the protocol version lists from the ACS Protocol Versions and the DS Protocol Versions obtained from the PRes message and the protocol version supported by the 3DS SDK to set the highest common Message Version Number.

The 3DS Requestor App ~~invokes~~ `createTransaction` ~~method within~~ **activates the 3DS SDK** to initiate 3-D Secure Cardholder authentication.

The second Note at the end of Step 2 was revised.

Note: As described in the applicable ~~EMV® 3-D Secure~~ 3DS SDK ~~Specification~~ specification, the 3DS SDK encrypts the Device Information by using the DS public key. This key is identified based on the Directory Server ID that is passed ~~to the createTransaction method~~ when the 3DS SDK is activated.

Step 4: The 3DS Requestor Environment

New Note was added directly after Requirement 1.

Note: For a Split-SDK refer to Section 4 of the *EMV 3-D Secure Split-SDK Specification*.

During the execution of this Step, the 3DS SDK shall:

[Req 2]

Obtain the Device Information, SDK Reference Number, and SDK App ID. Refer to the applicable ~~EMV 3-D Secure~~ 3DS SDK ~~Specification~~ specification and Annex A of this specification for additional detail.

Step 5: The 3DS Server

The 3DS Server shall:

~~**[Req 11]**~~

~~Determine which DS the authentication transaction needs to be sent based on the BIN (as defined in ISO 7812) and optionally other Cardholder account information.~~

[Req 12]

Establish a secure link with the DS as defined in Section 6.1.2.1.

If the connection cannot be established with the DS, the 3DS Server **ends 3-D Secure processing**.

If no PRes message information is available, then the 3DS Server may use a Message Version Number supported by the 3DS Server.

In the same Step, the Note following Requirement 14 was deleted.

~~**Note: The 3DS Server can use the ACS Start Protocol Version, ACS End Protocol Version, DS Start Protocol Version and DS End Protocol Version obtained from the PRes message to verify that the ACS and DS support the protocol version used by the 3DS Server. In addition, the 3DS Server can use the ACS Information Indicator to identify the features that the Account Range supports (for example, Decoupled Authentication and/or Whitelisting).**~~

Step 6: The DS

The DS shall:

[Req 15]

Receive the AReq message from the 3DS Server and Validate as defined in Section 5.9.1.

If the message is in error, the DS **ends processing**.

[Req 390]

If SDK Type = 02, verify the signature in the SDK Server Signed Content, as defined in Section 6.2.2.4.

If the verification fails, the DS returns to the 3DS Server an Error Message (as defined in Section A.9) with Error Component = D and Error Code = 308 and **ends processing**.

[Req 394]

Determine if the SDK Type and the SDK Reference Number are valid for the transaction according to DS rules.

If not, the DS returns to the 3DS Server an Error Message (as defined in Section A.9) with Error Component = D and Error Code = 305 and **ends processing**.

[Req 16]

Generate the DS Transaction ID.

[Req 420]

Identify the DS public key used by the 3DS SDK to encrypt Device Information from the key identifier (kid) in the SDK Encrypted Data.

If the DS detects an error with the key identifier, the DS returns to the 3DS Server an Error Message (as defined in Section A.9) with Error Component = D and Error Code = 311 and **ends processing**.

[Req 17]

If decryption fails, the DS returns to the 3DS Server an Error Message (as defined in Section ~~A5.5~~A.9) with Error Component = D and Error Code = 302 and **ends processing.**

[Req 18]

If not, the DS returns to the 3DS Server EITHER:

- an Error Message (as defined in Section ~~A.5.5~~A.9) with Error Component = D and Error Code = 102 and **ends processing**.

[Req 19]

- If either:
  - the 3DS Server Reference Number does not represent a participating 3DS Server, OR
  - the SDK Reference Number does not represent a participating 3DS SDK

  then the DS returns to the 3DS Server an Error Message (as defined in Section ~~A.5.5~~A.9) with Error Component = D and Error Code = 303 and **ends processing**.
- If Merchant Category Code (MCC) is not valid for the specific DS, then the DS returns to the 3DS Server Error Message (as defined in Section ~~A.5.5~~A.9) with Error Component = D and Error Code = 306 and **ends processing**.

Step 7: The ACS

The ACS shall:

[Req 386]

Check whether the ~~SDK~~ Device Information ~~Data Version Number~~ data elements correspond to the Data Version Number~~is recognised~~.

~~If not recognised, the ACS proceeds with processing the transaction and does not error due to the unrecognised Data Version Number.~~ If the Device Information data elements do not match the Data Version, the ACS returns an error message (Error Code = 203) or proceeds to process the transaction.

New Note was added after Requirement 30.

Note: SPC authentication is not supported in the App-based authentication flow; therefore, Transaction Status = S is not allowed.

[Req 32]

If a challenge is deemed necessary (Transaction Status = C), the ACS determines whether an acceptable challenge method is supported by the 3DS SDK based in part on the following data elements received in the AReq message: Device Channel, Device Rendering Options Supported, ~~and~~ SDK Maximum Timeout and SDK Type. The ACS performs the following:

No change to bullets a–e.

[Req 321]

If a Decoupled Authentication challenge is deemed necessary (Transaction Status = D) and 3DS Requestor Decoupled Request Indicator = Y or B, the ACS determines whether an acceptable challenge method is supported by the ACS based in part on the following data element received in the AReq message: 3DS Requestor Decoupled Max Time. The ACS performs the following:

b. Includes Decoupled (= 12) in Authentication Method.

Subsequent bullets were renumbered accordingly.

Step 8: The DS

The DS shall:

[Req 305]

Check the data elements in the ARes message as follows:

- If the ACS Reference Number does not represent a participating ACS, the DS shall:
  - Return to the 3DS Server an Error Message (as defined in Section ~~A.5.5~~A.9) with Error Component = D and Error Code = 303 and **ends processing**, OR
  - Sends an ARes message (as defined in Table B.2) to the 3DS Server with Transaction Status set to the appropriate response as defined by the specific DS.

New Requirement 421 was added directly after Requirement 36.

[Req 421]

If the DS creates the ARes message on the ACS's behalf (for example, the DS returns a Transaction Status = A), then the DS sets the ACS Reference Number equal to the DS Reference Number and the ACS Transaction ID equal to the DS Transaction ID and all the other ACS data elements required in the ARes message according to the DS capabilities.

Step 9: The 3DS Server

[Req 355]

~~If~~ Convey the Cardholder Information Text to the 3DS Requestor Environment. The 3DS Requestor displays the Cardholder Information Text received to the Cardholder as depicted in Section A.20~~has been provided by the ACS for this transaction the 3DS Server shall ensure the Cardholder Information Text is displayed on the 3DS Requestor App~~.

Step 10: The 3DS Requestor App

The 3DS Requestor App ~~invokes the "doChallenge method"~~ **performs the challenge** by making a call to the 3DS SDK. ~~Refer to the EMV 3-D Secure SDK Specification for additional information about this method.~~

Step 14: The 3DS SDK

[Req 55]

Display the UI based upon the ACS UI Type selected and the data elements populated. Refer to Section 4.2 ~~and refer to the *EMV 3-D Secure – SDK Specification*~~ of the applicable 3DS SDK specification or for an OOB authentication (ACS UI Type = 04 or 06) refer to Section 3.2 for UI details.

Step 15: The Cardholder Interaction with the 3DS SDK

Requirement 59 was moved from Step 16 to Step 15.

[Req 59]

If the Cardholder abandons the challenge during the processing of Step 12 through Step 15, the 3DS SDK sets the Challenge Cancelation Indicator to the appropriate value in the CReq message ~~and sends the CReq message to the ACS using the secure link established in [Req 56]~~.

Note: For ACS UI Type = 04 or 06, see Section 3.2 for OOB authentication requirements.

Step 16: The 3DS SDK

Requirement 59 was moved from Step 16 to Step 15.

The 3DS shall:

[Req 58]

Send ~~the~~ one CReq message to the ACS using the secure link established in **[Req 56]** and wait for the ACS CRes message response before sending another CReq message.

Step 17: The ACS

New Requirement 461 was added after Requirement 310.

[Req 461]

If the ACS determines that Decoupled Authentication Fallback is necessary and 3DS Requestor Decoupled Request Indicator = F or B, inform the Cardholder of Decoupled Authentication using the Information UI template as defined in Chapter 4 with additional CRes messages, then continue to **[Req 61]** to prepare the final CRes message.

[Req 61]

Check the ~~authentication~~ data ~~entered by the Cardholder:~~received in the CReq message and assess the status of the authentication.

- If the ACS selects Decoupled Authentication Fallback, then the ACS continues with Step 18.

- If ~~correct~~the authentication is successful, then the ACS:
  - Increments the Interaction Counter
  - Sets ~~the~~ Transaction Status = Y
  - Sets the ECI value as defined by the specific DS
  - Generates the Authentication Value as defined by the DS
  - Sets ~~the~~ Challenge Completion Indicator = Y
  - Continues with Step 18
- If ~~incorrect and~~the authentication has failed or is not completed, then the ACS:
  - Increments the Interaction Counter and compares it to the ACS maximum challenges.
  - If the Interaction Counter ≥ ACS maximum challenges or the authentication has failed, the ACS:
    - Sets ~~the~~ Transaction Status = N
    - Sets ~~the~~ Transaction Status Reason = 19
    - Sets the ECI value as defined by the specific DS
    - Sets ~~the~~ Challenge Completion Indicator = Y
    - Continues with Step 18
  - Else if~~If~~ the Interaction Counter < ACS maximum challenges and the authentication is not completed, the ACS:
    - Obtains the information needed to display a repeat Challenge on the Consumer~~'s~~ Device per the selected challenge method and ACS UI Type.
    - Continues with Step 13.

Step 18: The ACS

New Requirement 462 was added directly before Requirement 62.

[Req 462]

For a Challenge Flow (ARes Transaction Status = C), if 3DS Requestor Decoupled Request Indicator = F or B, and if the ACS has determined that Decoupled Authentication Fallback is necessary,

- Set Transaction Status = D.
- Set Transaction Status Reason = 29 or 30.

Step 20: The 3DS Server

New Requirement 463 was added directly after Requirement 70.

The 3DS Server shall:

[Req 463]

If Transaction Status = D and if 3DS Requestor Decoupled Request Indicator = F or B, set Results Message Status to 04.

Note: The 3DS Server initiates a 3RI transaction with Decoupled Authentication as defined in Section 3.4.

Step 23: The ACS

The ACS shall for a Challenge Flow transaction (ARes Transaction Status = C) as a continuation of receiving the CReq message in Step 17, do the following:

[Req 470]

If 3DS Requestor Decoupled Request Indicator = F or B and if Transaction Status = D in the RReq message, set Transaction Status to D in the Final CRes message.

Requirement 76 follows unchanged.

3.2 Challenge Flow with OOB Authentication Requirements

Unchanged text in this section is provided for context.

An Out-of-Band (OOB) Challenge Flow is identical to a standard 3-D Secure Processing Flow as defined in Section 3.1 with the following exceptions:

Step ~~6~~7: The ACS recognises that an OOB interaction with the Cardholder is required.

Step 13: The challenge information in the CRes message consists of Cardholder instructions on how to perform the OOB authentication.

Between Step ~~12~~13 and Step 15: ~~Between Step 12 and Step 15 t~~The ACS initiates an OOB interaction with the Cardholder rather than interacting with the Cardholder via the 3DS SDK. During the OOB authentication the Cardholder authenticates to the ACS or a service provider/Issuer interacting with the ACS. See Section 3.2.1 for additional OOB requirements.

The method used for the OOB communication and the authentication method itself is outside the scope of this specification. An example of an OOB communication could be a push notification to a banking app that completes authentication and then sends the results to the ACS.

The ACS may use a combination of OOB automatic switching options (OOB App URL, 3DS Requestor App URL) to switch between the 3DS Requestor App and the OOB Authentication App following the requirements in Section 3.2.2.

~~**Step 13:** The challenge information in the CRes message consists only of Cardholder instructions on how to perform the OOB authentication.~~

Step ~~16~~17: The ACS receives only an acknowledgement that the Cardholder may have performed the OOB authentication~~. **Step 17:**~~, thus ~~I~~in **[Req 61]** ~~of the Challenge Flow,~~ the ACS gathers the information on whether the authentication was successful from the OOB interaction with the Cardholder ~~rather than from~~instead of the CReq message. If the ACS determines that the Cardholder did not authenticate, ~~it~~then the ACS can update the Cardholder instructions through another CRes message.

~~When a Cardholder returns to the 3DS Requestor App from an OOB app on the same device, further Cardholder action will not be required to complete the authentication. In this scenario, the SDK will automatically post a CReq message when the 3DS Requestor App is moved to the foreground.~~

How an authentication decision is made for an OOB authentication is outside the scope of this specification. However, the ACS needs access to the result of the OOB authentication before Step 18.

~~**Note: The 3DS Requestor should consider that an OOB authentication can take longer for the Cardholder to complete and therefore should adjust the 3DS SDK's challenge time-out accordingly.**~~

Note: OOB authentication as defined in this section is a separate authentication flow from the Decoupled Authentication flow.

The requirements defined in this Section 3.2 describe the additional flow and requirements specific to ACS UI Type 04 and 06.

3.2.1 OOB Requirements

This section defines additional requirements for an OOB flow.

Step 15: The Cardholder Interaction with the 3DS SDK

The 3DS SDK shall:

[Req 399]

For ACS UI Type = 04, set the OOB Continuation Indicator = 01 when the Cardholder selects the button with the OOB Continuation Label.

[Req 400]

For ACS UI Type = 04 or 06, if the 3DS Requestor App comes to the foreground (for example, the Cardholder returns to the 3DS Requestor App that comes to the foreground), set the value of the OOB Continuation Indicator = 02, and continue automatically (without UI interaction by the Cardholder) with Step 16 in the App-based flow (send a CReq message to the ACS).

3.2.2 OOB Automatic Switching Features

This specification defines the following automatic switching features for an OOB authentication:

- The OOB App URL (in the CRes message) that the 3DS SDK uses to automatically switch to the OOB Authentication App when the Cardholder chooses to transfer control.
- The 3DS Requestor App URL (in the CReq message) that the OOB Authentication App uses to automatically transfer control to the 3DS Requestor App when the OOB Authentication App has concluded the Cardholder interaction.

To accommodate error scenarios, when an automatic switching between the two apps is unsuccessful, the 3DS SDK automatically sends a CReq message (once the 3DS Requestor App has returned to the foreground) so that the ACS understands the latest status of the authentication attempt. The ACS may then choose next authentication steps dependent on whether an authentication via the OOB Authentication App occurred.

Note that when an OOB Authentication App is on a different device than the 3DS Requestor App, then automatic switching is not possible and the Cardholder must manually switch to the app on a secondary device.

The following additional requirements apply if an automatic switching feature is utilised.

Step 13: The ACS

Before **[Req 51]**, the ACS additionally prepares the CRes message for an OOB authentication.

If using the OOB App URL feature, the ACS shall:

[Req 401]

For ACS UI Type = 04 or 06, set the OOB App URL to the URL value used during installation of the OOB Authentication App.

[Req 402]

For ACS UI Type = 06, include in the OOB challenge HTML code an action that triggers a location change to the `HTTPS://EMV3DS/openoobApp` URL when the Cardholder selects the OOB App URL button.

Note: If the ACS includes additional actions (for example, Complete button) for the Cardholder in the HTML code, it uses the `HTTPS://EMV3DS/challenge` URL as defined in [Req 164].

Step 14: The 3DS SDK

If the OOB App URL is present in the CRes message, then the 3DS SDK performs an additional requirement for this step:

After having performed **[Req 54]**, as part of **[Req 55]**, the 3DS SDK shall:

[Req 403]

For ACS UI Type = 04, display a button with the OOB App Label used for the switch to the OOB Authentication App.

Step 15: The Cardholder Interaction with the 3DS SDK

The Cardholder interacts with the 3DS SDK User Interface (UI). For example, selects the OOB Continuation button or the OOB App Label button.

3.2.2.1 OOB App URL Requirements

If the OOB App URL is present in the CRes message, then the 3DS SDK shall:

[Req 472]

Check that the OOB App URL uses the HTTPS scheme.

If not, the 3DS SDK returns an error as defined in Section 5.9.7.

[Req 404]

For ACS UI Type = 04, attempt to open the OOB Authentication App by using the OOB App URL when the Cardholder selects the button with the OOB App Label.

[Req 405]

For ACS UI Type = 06:

a. Intercept a location change event that is sent to the specific `HTTPS://EMV3DS/openoobApp` URL.

b. Attempt to open the OOB Authentication App by using the OOB App URL.

[Req 406]

If the attempt to open the OOB Authentication App is successful, then continue with Section 3.2.2.2.

[Req 407]

If the attempt to open the OOB Authentication App fails (for example, the platform method to open the OOB App URL returns an error), then:

a. Set the OOB App Status to the appropriate value as defined in Table A.1.

b. Set the OOB Continuation Indicator = 02.

c. Continue automatically (without UI interaction by the Cardholder) with Step 16 in the App flow.

Note: If the OOB Authentication App is not present on the device, then the device Operating System (OS) attempts to open the OOB App URL using the device's default browser. The Issuer provides a web page for this URL to instruct the Cardholder on how to manually perform the OOB Authentication App switch.

3.2.2.2 3DS Requestor App URL

When the OOB Authentication App invokes the 3DS Requestor App URL, the device OS switches to the 3DS Requestor App that moves to the foreground. The 3DS Requestor App transfers the control back to the 3DS SDK.

[Req 475]

When receiving the CReq message, the ACS shall check that the 3DS Requestor App URL uses the HTTPS scheme.

If not, the ACS returns an error as defined in Section 5.9.5.

If the 3DS Requestor App URL is available to the 3DS SDK, then the following requirements are performed.

The 3DS SDK shall:

[Req 408]

Display the UI template and data elements received in the last CRes message.

[Req 409]

For ACS UI Type = 04 or 06, set the OOB Continuation Indicator = 02 and continue with Step 16 in the App-based flow.

3.3 Browser-based Requirements

Step 2: The 3DS Server/3DS Requestor

The 3DS Requestor uses the Cardholder Account Number and optionally other Cardholder information to request the ACS ~~Start~~ Protocol Version~~, ACS End Protocol Version, DS Start Protocol Version~~ and DS ~~End~~ Protocol Version lists and if present, the 3DS Method URL for that ~~BIN~~ card range from the 3DS Server.

The 3DS Server shall:

[Req 80]

Retrieve the ACS ~~Start~~ Protocol Version and ~~ACS End Protocol Version, DS Start Protocol Version and~~ DS ~~End~~ Protocol Version lists and, if present, the 3DS Method URL (stored from a previously received PRes message) for ~~that BIN~~ the card range.

Requirement 81 remains unchanged.

[Req 82]

Pass the 3DS Server Transaction ID, ACS ~~Start~~ Protocol Versions, ~~ACS End Protocol Version, DS Start Protocol Version,~~ DS ~~End~~ Protocol Versions and if present, the 3DS Method URL back through the 3DS Requestor Environment to the 3DS Requestor.

If the DS ~~Start~~ Protocol Versions ~~and DS End Protocol Version are~~ is not present for the ~~BIN~~ card range, then the default values for the DS ~~Start~~ Protocol Versions ~~and DS End Protocol Version~~ located in the PRes message shall be utilized.

Step 3: The 3DS Requestor Environment

The 3DS Server shall:

[Req 84]

Ensure that the 3DS Method is executed on the 3DS Requestor Website if a 3DS Method URL exists ~~for this transaction~~as defined in Section 5.8.1.

Step 6: The 3DS Server

New Requirement 441 was added directly after Requirement 88.

The 3DS Server shall:

[Req 441]

Ensure that the 3DS Requestor executed the 3DS Method within the previous 10 minutes. Otherwise, the 3DS Requestor re-executes the 3DS Method as defined in Section 5.8.1.

New Requirement 422 was added directly after Requirement 90.

[Req 422]

Use the protocol version lists from the ACS Protocol Versions and DS Protocol Versions obtained from the PRes message to set the highest common Message Version Number.

If no PRes message information is available, then the 3DS Server may use a Message Version Number supported by the 3DS Server.

Note following Requirement 92 was deleted.

~~**Note: The 3DS Server can use the ACS Start Protocol Version, ACS End Protocol Version, DS Start Protocol Version and DS End Protocol Version obtained from the PRes message to verify that the ACS and DS support the protocol version used by the 3DS Server. In addition, the 3DS Server can use the ACS Information Indicator to identify the features that the Account Range supports (for example, Decoupled Authentication and/or Whitelisting).**~~

Step 7: The DS

References in Requirement 95 and Requirement 96 to Section A.5.5 were replaced with references to Section A.9.

Step 8: The ACS

The ACS shall:

Requirements 102 and 103 remain unchanged.

Note: The ACS uses the ~~Device~~ Browser Information (as defined in Section A.6) received in the AReq message and the 3DS Method to recognise the device, assess transaction risk, and determine if it can complete the authentication. When Decoupled Authentication is used, the Consumer Device that initiated the transaction does not need to be supported when the ACS has alternative approaches to authenticating the Cardholder.

[Req 410]

Retrieve the data from a previous 3DS Method execution if the 3DS Method ID is present.

The ACS shall:

[Req 107]

New bullet was added at the end:

- requiring an SPC authentication (Transaction Status = S). See Section 3.5.1 for details.

[Req 325]

If a Decoupled Authentication challenge is deemed necessary (Transaction Status = D) and 3DS Requestor Decoupled Request Indicator = Y or B, the ACS determines whether an acceptable challenge method is supported by the ACS based in part on the following data element received in the AReq message: 3DS Requestor Decoupled Max Time.

The remainder of the requirement follows unchanged.

Step 9: The DS

References in Requirement 306 to Section A.5.5 were replaced with references to Section A.9.

New Requirement 411 was added directly after Requirement 113.

The DS shall:

[Req 411]

If the DS creates the ARes message on the ACS's behalf (for example, the DS returns a Transaction Status = A), then set the ACS Reference Number equal to the DS Reference Number and the ACS Transaction ID equal to the DS Transaction ID and all the other ACS data elements required in the ARes message according to the DS capabilities.

Step 10: The 3DS Server

New Note following Requirement 327.

Note: For a 3DS Requestor-initiated SPC transaction (Transaction Status = S), see Section 3.5, Step 10).

[Req 356]

Convey ~~If~~ the Cardholder Information Text to the 3DS Requestor Environment. The 3DS Requestor displays the Cardholder Information Text received to the Cardholder as depicted in Section A.20~~has been provided by the ACS for this transaction the 3DS Server shall ensure the Cardholder Information Text is displayed on the 3DS Requestor website~~.

Step 12: The ACS and Browser

New Note following Requirement 122.

Note: An Out-of-Band (OOB) Challenge Flow is identical to a standard 3-D Secure Processing Flow for a challenge for the Browser channel.
The ACS UI consists of Cardholder instructions on how to perform the OOB authentication.
The ACS initiates an OOB interaction with the Cardholder rather than interacting with the Cardholder via the Browser challenge iframe. During the OOB authentication the Cardholder authenticates to the ACS or a service provider/Issuer interacting with the ACS.
The method used for the OOB communication and the authentication method itself is outside the scope of this specification. An example of an OOB communication could be a link to a banking website that completes authentication and then sends the results to the ACS.

Step 13: The Cardholder

The Cardholder interacts with the UI provided by the ACS and, if requested, enters the authentication data ~~as required by the ACS UI~~.

Step 14: The Browser

The Browser sends the entered ~~authentication~~ data to the ACS over the channel established by the HTTP POST in Step 10.

Step 15: The ACS

New Requirement 464 was added at the beginning of Step 15, directly before Requirement 123.

The ACS shall:

[Req 464]

If the ACS determines that Decoupled Authentication Fallback is necessary and 3DS Requestor Decoupled Request Indicator = F or B, inform the Cardholder of Decoupled Authentication before the final CRes message using the Information UI template as defined in Chapter 4.

[Req 123]

Check the authentication data ~~entered by the Cardholder~~received and assess the status of the authentication:

- If the ACS selects Decoupled Authentication Fallback, then the ACS continues with Step 16.
- If correct, then the ACS:
  - Increments the Interaction Counter
  - Sets ~~the~~ Transaction Status = Y
  - Sets the ECI value as defined by the specific DS
  - Generates the Authentication Value as defined by the DS
  - Continues with Step 16
- If ~~incorrect and~~ the authentication has failed, is not completed or the Cardholder has selected to cancel the authentication, then the ACS:
  - Increments the Interaction Counter and compares it to the ACS maximum challenges
  - If the Interaction Counter ≥ ACS maximum challenges or the authentication has failed or the Cardholder has selected to cancel the authentication, the ACS:
    - Sets ~~the~~ Transaction Status = N
    - Sets ~~the~~ Transaction Status Reason = 19
    - Sets the ECI value as defined by the specific DS
    - Continues with Step 16
  - Else ~~(~~if the Interaction Counter < ACS maximum challenges or the authentication is not completed, the ACS:
    - Obtains the information needed to display a repeat Challenge on the Consumer~~'s~~ Device per the selected challenge method and ACS UI Type.
    - Prepares the authentication User Interface (ACS UI) to the Cardholder Browser which may contain HTML, JavaScript, etc.
    - Continues with Step 12.

The process of exchanging HTML will repeat until a determination is made by the ACS.

Step 16: The ACS

New Requirement 465 was added directly before Requirement 124.

[Req 465]

For a Challenge Flow (ARes Transaction Status = C), if 3DS Requestor Decoupled Request Indicator = F or B, and if the ACS has determined that Decoupled Authentication Fallback is necessary,

- Set Transaction Status = D.
- Set Transaction Status Reason = 29 or 30.

Step 18: The 3DS Server

The 3DS Server shall:

New Requirement 466 was added directly after Requirement 132.

[Req 466]

If Transaction Status = D and if 3DS Requestor Decoupled Request Indicator = F or B, set Results Message Status to 04.

Note: The 3DS Server initiates a 3RI transaction with Decoupled Authentication as defined in Section 3.4.

Step 21: The ACS

The ACS shall, for a Challenge Flow transaction (ARes Transaction Status = C), as a continuation of receiving the CReq message in Step 11, do the following:

[Req 471]

If 3DS Requestor Decoupled Request Indicator = F or B and if Transaction Status = D in the RReq message, set Transaction Status to D in the Final CRes message.

[Req 138]

Format the Final CRes message as defined in Table B.5.

[Req 139]

Base64url-encode the Final CRes message and include, if present in the CReq message, the 3DS Requestor Session Data in HTML form (as defined in Table A.3).

3.4 3RI-based Requirements

Step 2: The 3DS Server

New Requirements 423 and 467 (including text from the Note below Requirement 277) were added after Requirement 275.

The 3DS Server shall:

[Req 423]

Use the protocol version lists from the ACS Protocol Versions and DS Protocol Versions obtained from the PRes message to set the highest common Message Version Number.

If no PRes message information is available, then the 3DS Server may use a Message Version Number supported by the 3DS Server.

[Req 467]

In the case of Decoupled Authentication Fallback, the 3DS Server initiates a 3RI authentication within 60 seconds of receiving the RReq message from the previous transaction, containing:

- 3RI Indicator = 19 (Decoupled Authentication Fallback)
- 3DS Requestor Decoupled Request Indicator = Y
- 3DS Requestor Prior Transaction Authentication Information object:
    - 3DS Requestor Prior Transaction Reference = ACS Transaction ID from the RReq message indicating that Decoupled Authentication is to be performed
    - 3DS Requestor Prior Transaction Authentication Method = 02 (Cardholder challenge occurred by ACS).

Note following Requirement 277 was deleted.

~~**Note: The 3DS Server can use the ACS Start Protocol Version, ACS End Protocol Version, DS Start Protocol Version and DS End Protocol Version obtained from the PRes message to verify that the ACS and DS support the protocol version used by the 3DS Server. If no PRes message information is available, then the 3DS Server may send an AReq message for all Cardholder accounts. In addition, the 3DS Server can use the ACS Information Indicator to identify the features that the Account Range supports (for example, Decoupled Authentication and/or Whitelisting).**~~

Step 3: The DS

References in Requirements 280 and 281 to Section A.5.5 were replaced with references to Section A.9.

New Requirement 427 was added directly after Requirement 283.

The DS shall:

[Req 427]

Store the 3DS Server URL with the DS Transaction ID (for possible RReq message processing).

Step 4: The ACS

The ACS shall:

[Req 291]

- ~~authentication not requested by the 3DS Server for~~ data sent for informational purposes only, an authentication not requested by the 3DS Server (Transaction Status = I)

Step 5: The DS

A reference in Requirement 308 to Section A.5.5 was replaced with a reference to Section A.9.

New Requirement 412 was added directly after Requirement 297.

The DS shall:

[Req 412]

If the DS creates the ARes message on the ACS's behalf (for example, the DS returns a Transaction Status = A), then the DS sets the ACS Reference Number equal to the DS Reference Number and the ACS Transaction ID equal to the DS Transaction ID and all the other ACS data elements required in the ARes message according to the DS capabilities.

3.5 SPC-based Authentication Requirements

Section 3.5 (including Sections 3.5.1 and 3.5.2) is an entirely new section and is not replicated in this specification bulletin.

Chapter 4 EMV 3-D Secure User Interface Templates, Requirements and Guidelines

Chapter 4 includes both new and updated figures that are not depicted in this Specification Bulletin. Figure numbers and references are updated as applicable.

4.1 3-D Secure User Interface Templates

Note added directly below Figure 4.1:

Note: The landscape UI layout is defined to provide a consistent user experience across all consumer devices or ecosystems. For example:

- **Large devices where landscape mode is the only display mode (i.e., large television screens).**
- **Small devices where both portrait and landscape modes are available (i.e., mobile, tablets and PCs).**

The 3DS SDK shall:

[Req 395]

Support the UI template orientation(s) (i.e., portrait and landscape) according to the ~~device capabilities~~ App Screen Orientation.

In Requirement 358, a reference to Table A.18 was replaced with a reference to Table A.20.

New Requirements 418 and 391 were added directly after Requirement 358.

[Req 418]

Support full-screen vertical and horizontal scrolling for HTML UI, and at minimum, support full-screen vertical scrolling for the Native UI, for all ACS-provided content. The Header zone may remain anchored at the top of the display.

[Req 391]

Ensure that the Header zone for the UI does not occupy more than 10 percent of the screen height.

The ACS shall:

[Req 342]

Support all ACS Rendering Types for the ACS supported authentication methods, at a minimum at least one ACS UI Template for each ACS Interface in addition to the Information UI template.

The text directly following Requirement 359 was updated as follows:

Figure 4.3 ~~and Figure 4.4~~ through Figure 4.5: illustrate the consistency of the look and feel across device channels and implementations.

The text preceding Figure 4.5 was updated as follows:

Figure 4.5 depicts sample Native UI formats with or without a device header.

Figure 4.5: Sample ~~Native~~ UI OTP/Text Template with/without Device Header

4.2 App-based User interface Overview

The supported digital image file types are png and jpeg ~~tiff and bmp~~. Any other image types implemented by the ACS may not be supported by the 3DS SDK.

4.2.1.1 3DS SDK/3DS Requestor App

[Req 142]

Not include any other design element or text in the Processing screen.

[Req 145]

Display the Processing screen supplied by the 3DS SDK during the entire AReq/ARes message cycle and overlay on the merchant checkout page ~~including the overlay the Header Zone~~ as depicted in Figure 4.7 and Figure 4.8.

[Req 147]

Create the Processing screen with only the default Processing Graphic (for example, a progress bar or a spinning wheel) of the Consumer Device OS ~~(See Figure 4.5 and Figure 4.6)~~ without words, text or white box (See Figure 4.9–Figure 4.12).

Text directly preceding Figure 4.9 was amended as follows:

Figure 4.9 provides a sample format for the App-based processing flow. ~~Figure 4.10 provides a sample format for the Out of Band template and 3DS Requestor App on the same device for an App-based processing flow. Figure 4.11 provides a sample format for the Decoupled Authentication Flow.~~

Text directly preceding Figure 4.10 reads as follows:

Figure 4.10 provides a sample format for the OOB template for a manual transfer to and from the OOB Authentication App for an App-based processing flow.

Figure 4.10 Sample OOB Template (~~OOB App and 3DS Requestor App on the same device~~Manual transfer to and from the OOB App)—App-based Processing Flow

Figure 4.11 provides a sample format for the OOB template for a manual transfer to the OOB App and an automatic return to the 3DS Requestor App for an App-based processing flow. The 3DS Requestor and OOB Apps are on the same device.

Figure 4.11 Sample OOB Template (OOB App and 3DS Requestor App on same device)—w/o OOB App launch button—App-based Processing Flow

Figure 4.12 provides a sample format for the OOB template for an automatic transfer to and from the OOB App for the App-based processing flow. The 3DS Requestor and OOB Apps are on the same device.

Figure 4.12 Sample OOB Template (OOB App and 3DS Requestor App on same device with OOB App launch button)—App-based Processing Flow

Figure 4.13 provides a sample format for the Decoupled Authentication Flow.

Figure 4.13 Sample Decoupled Authentication Template—App-based Processing Flow

4.2.2 Native UI Display Requirements

With the 3DS SDK's knowledge of the device screen size and orientation, font size, etc., the 3DS SDK can optimise the content provided by the issuer (for example, by removing an extra line feed that would cause scrolling).

4.2.2.1 3DS SDK/ACS

The 3DS SDK shall for the CReq/CRes message exchange:

[Req 362]

For the ACS UI Type and the ~~device screen orientation~~ App Screen Orientation, display all the ~~supported UI template in its applicable orientation and the supported~~ provided UI data elements in their applicable zones and order as defined in Table ~~A.18~~A.20 and depicted in Figure 4.1 and Figure 4.2. The expected format is depicted in Sections 4.2.3 and 4.2.6.

If the 3DS SDK receives an unsupported UI data element(s) for this ACS UI Type, the 3DS SDK does not display the UI data elements, proceeds with the challenge and does not send an Error Message to the ACS.

[Req 398]

For the ACS UI Type, the 3DS SDK returns to the ACS an Error Message (as defined in Section ~~5.5~~A.9) with Error Component = ~~S~~C and Error Code = 201 if any mandatory UI data elements are missing as defined in Table ~~A.18~~A.20.

[Req 366]

Display the Expandable Information Text only when the ~~user~~Cardholder selects the Expandable Information Label.

New Requirements 392 and 446 were added directly after Requirement 369.

[Req 392]

Display the Trust List Information Text with the default setting = Off so that a Cardholder action is required to generate a Y value in Trust List Data Entry.

[Req 446]

Display the Device Binding Information Text with the default setting = Off so that a Cardholder action is required to generate a Y value in Device Binding Data Entry.

The ACS shall for the CReq/CRes message exchange:

[Req 387]

Include only the ~~mandatory and the optional~~ ACS-chosen UI data elements supported for the selected ACS UI Type as defined in Table ~~A.18~~A.20.

[Req 370]

If ~~a carriage return is~~ used, ~~then~~ represent ~~the~~ a carriage return as specified in Table A.1 for the following data elements:

No edits were made to the bulleted list of this Requirement.

New Requirements 445 and 429 were added directly after Requirement 370.

[Req 445]

If used, represent a bold text as specified in Table A.1 for the following data elements:

- Challenge Information Text
- Expandable Information Text
- Why Information Text

[Req 429]

Include the image in either png or jpeg format when providing the Issuer Image and Payment System Image.

4.2.3 Native UI Templates

Figure ~~4.12~~4.14 through Figure ~~4.23~~4.37 depict sample Native UI templates.

Figure ~~4.12~~4.14 and Figure ~~4.13~~4.15 provide sample formats for a one-time passcode (OTP)/Text during a Payment Authentication transaction. This sample UI provides a format using expandable fields for additional information.

Figure ~~4.12~~4.14: Sample Native UI OTP/Text Template—PA—Portrait

Figure ~~4.13~~4.15: Sample Native UI OTP/Text Template—PA—Landscape

Figure 4.16 and Figure 4.17 provide sample formats with the optional second one-time passcode (OTP)/Text during a Payment Authentication transaction. This sample UI provides a format using expandable fields for additional information.

Figure 4.16: Sample Native UI with Optional Second OTP/Text entries Template—PA—Portrait

Figure 4.17: Sample Native UI with Optional Second OTP/Text entries Template—PA—Landscape

Figure ~~4.14~~4.18: Sample Native UI/OTP/Text Template—NPA

Figure ~~4.15~~4.19 and Figure ~~4.16~~4.20 provide sample formats that allow multiple options to be presented to the Cardholder to obtain a single response.

Figure ~~4.15~~4.19: Sample Native UI—Single-select Information—PA—Portrait

Figure ~~4.16~~4.20: Sample Native UI—Single-select Information—PA—Landscape

Figure ~~4.17~~4.21 and Figure ~~4.18~~4.22 provide sample formats that allow multiple options to be presented to the Cardholder to obtain multiple responses on a single screen.

Figure ~~4.19~~4.23 and Figure 4.26 provide~~s~~ sample OOB formats to display instructions to the Cardholder.

Figure ~~4.19~~4.23 Sample OOB Native UI Template with Complete button—PA—Portrait

Figure ~~4.20~~4.24: Sample OOB Native UI Template with Complete button—PA—Landscape

Figure 4.25 and Figure 4.26 provide sample OOB formats that display a button to open an Authentication App.

Figure 4.25: Sample OOB Native UI Template with Automatic OOB App URL link—Portrait

Figure 4.26: Sample OOB Native UI Template with Automatic OOB App URL link—Landscape

Figure ~~4.21~~4.27: Sample Challenge Information Text Indicator—PA

Figure 4.28–Figure 4.33 provide sample formats that display the ~~Whitelisting~~ two possible positions for the Trust List option ~~to~~ and Device Binding option (above or below the buttons) as defined in Table A.20.

Note: The sample format depicts the UI after the Cardholder ~~during a purchase authentication~~has entered the OTP and selected the Trust List and/or the Device Binding checkbox or switches.

Figure ~~4.22~~4.28: Sample ~~Whitelisting~~Trust List/Device Binding Information Text—PA—Portrait

Figure ~~4.23~~4.29: Sample ~~Whitelisting~~ Trust List/Device Binding Information Text—PA—~~Landscape~~Portrait

Figure 4.30: Sample Trust List/Device Binding Information Text—PA—Landscape

Figure 4.31: Sample Trust List/Device Binding Information Text—PA—Landscape

Figure 4.32 and Figure 4.33 provide sample UI formats to display instructions to the Cardholder.

The Information user interface allows Issuers to display specific information during the challenge, for example to recover from an error situation or for the Cardholder to confirm consent.

Figure 4.32: Sample Information Native UI Template—PA—Portrait

Figure 4.33: Sample Information Native UI Template—PA—Landscape

Figure 4.34 Figure 4.35 provide a sample format that uses the Challenge Data Entry Masking and Challenge Data Entry Masking Toggle options during a purchase authentication.

Note: The sample format depicts the UI after the Cardholder enters the OTP and selects the Challenge Data Entry Masking Toggle.

Figure 4.34: Sample Challenge Data Entry Masking—PA

Figure 4.35: Sample Data Entry Masking with Toggle

Figure 4.36 and Figure 4.37 provide sample formats with the Challenge Additional Label button, which provides additional flexibility in challenge management available for all UI templates.

Figure 4.36: Sample Native UI OTP/Text Template with Challenge Additional Label—PA—Portrait

Figure 4.37: Sample Native UI OTP/Text Template with Challenge Additional Label—PA—Landscape

4.2.4 Native UI Message Exchange Requirements

4.2.4.3 3DS SDK

The 3DS SDK shall:

[Req 154]

Act upon any action to exit the 3DS SDK (for example, the Cancel action on screen or through an external controller) and return control to the 3DS Requestor App.

New Requirement 473 (originally Requirement 71 in the EMV® 3-D Secure SDK Specification, which has now been deleted) was added directly after Requirement 158.

[Req 473]

If the Cardholder does not enter any data in the UI, send the Challenge No Entry field with the value = Y in the CReq message.

4.2.5 HTML UI Display Requirements

The 3DS SDK will display the HTML ~~exactly~~ as provided by the Issuer. As such, it is the Issuer's responsibility to format the HTML to best display on the Consumer Device. Unlike the Native UI where the 3DS SDK can adjust the content provided by the Issuer, the HTML provided by the Issuer will be ~~exactly~~ what is displayed to the Cardholder.

Details of the HTML UI and the rendering process are separately described in the ~~EMV 3-D Secure~~ applicable 3DS SDK ~~Specification~~specification and in the documentation provided by each DS.

4.2.5.1 3DS SDK/ACS

Amended introduction to Requirements 375–378.

The ACS shall, if ~~using~~providing values for the form elements corresponding to the following ~~optional~~ data elements, provide the:

[Req 376]

Expandable Information Text for display in the Information zone only when the ~~user~~ Cardholder selects the Expandable Information Label.

Requirement 377 follows unchanged.

[Req 378]

Why Information Text for display in the Information zone only when the ~~user~~ Cardholder selects the Why Information Label.

4.2.6 HTML UI Templates

Figures in this section were renumbered.

Figure 4.41: Sample OOB HTML UI Template with Complete button—PA—Portrait

Figure 4.42: Sample OOB HTML UI Template with Complete button—PA—Landscape

Figure 4.43: Sample OOB HTML UI Template with OOB App URL button—PA—Portrait

Figure 4.44: Sample OOB HTML UI Template with OOB App URL button—PA—Landscape

Figure 4.45 and Figure 4.46 provide sample HTML Information UI templates to display instructions to the Cardholder that includes Issuer branding. The Information user interface allows Issuers to display specific information during the challenge, for example to recover from an error situation or for the Cardholder to confirm consent.

Figure 4.45: Sample Information HTML UI Template—Portrait

Figure 4.46: Sample Information HTML UI Template—Landscape

4.2.7 HTML Message Exchange Requirements

4.2.7.2 ACS

[Req 164]

Include in the ACS HTML an action which triggers a location change to a specified (HTTPS://EMV3DS/challenge) URL upon the Cardholder completing data input and pressing Submit, for ACS UI Type = 05, and only if manual app switching is used for OOB for ACS UI Type = 06.

4.2.7.3 3DS SDK

The 3DS SDK shall:

[Req 171]

Return control to the 3DS Requestor App when the Cancel action is selected.

On HTML submit, 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 URL (`HTTPS://EMV3DS/challenge`) with the challenge responses appended to the URL.

~~On HTML submit:~~

- ~~The web view will return, either a parameter string (HTML Action = GET) containing the cardholder's data input.~~

The second bullet of Requirement 171 is now a new separate Requirement 393, added after new Requirement 413.

[Req 413]

Monitor the URL changes, to retrieve the Cardholder response as query parameters from the URL (`HTTPS://EMV3DS/challenge`) and return a parameter string (HTML Action = GET) containing the Cardholder data input.

[Req 393]

Pass the received data input, unchanged, to the ACS in the Challenge HTML Data Entry data element of the CReq message. The 3DS SDK shall not modify or reformat this received data input.

New Requirement 474 (originally Requirement 75 in the EMV® 3-D Secure SDK Specification, which has now been deleted) was added directly after Requirement 393.

[Req 474]

When the Cardholder submits their response, if the 3DS SDK receives a blank response, then it assumes that the HTML is not valid. In this event, the 3DS SDK shall return to the ACS an Error Message (as defined in Section A.9) with Error Component = C and Error Code = 203.

The 3DS SDK transmits the CReq message to the ACS.

New example at the end of Section 4.2.7.3.

Example Cardholder Response

`HTTPS://EMV3DS/challenge?response=12348&submit=verify`, the 3DS SDK returns the cardholder data input `"challengeHTMLDataEntry" : "response=1234&submit=verify"`

4.3 Browser-based User Interface Overview

4.3.1 Processing Screen Requirements

4.3.1.1 3DS Requestor Website

The 3DS Requestor shall:

[Req 173]

Display the Processing screen that conveys to the Cardholder that processing is occurring (Refer to Figure ~~4.20~~4.50 and Figure ~~4.21~~4.51 for examples).

[Req 174]

Include the DS logo for display with or without a white box at the centre of the screen unless specifically requested not to include.

[Req 175]

Not include any other design element or text in the Processing screen.

4.3.1.2 ACS

The ACS shall:

[Req 177]

Create and maintain versions of the HTML that correspond to the sizes of the Challenge Window Size data element as defined in Table A.1 and provide the appropriate size in the CRes message based upon the Challenge Window Size that was provided by the 3DS Server in the ~~AReq~~CReq message.

[Req 178]

Create a Processing screen without words, text or white box for display during the HTML exchange CReq/CRes message cycle.

Requirement 181 was moved from its original position after Requirement 180 to a new position – directly after Requirement 178.

[Req 181]

Not include the DS logo or any other design element in the Processing screen.

[Req 180]

Include the DS logo in the HTML for display in the Branding Zone ~~for display during the challenge flow, (with the exception of the Processing screen)~~ unless specifically requested not to include.

4.3.2 Browser Display Requirements

4.3.2.1 ACS

Amended introduction to Requirements 381 through 384.

The ACS shall, if ~~using~~ providing values for the form elements corresponding to the following ~~optional~~ data elements, provide the:

[Req 382]

Expandable Information Text for display in the Information zone only when the ~~user~~ Cardholder selects the Expandable Information Label.

Requirement 383 follows unchanged.

[Req 384]

Why Information Text for display in the Information zone when the ~~user~~ Cardholder selects the Why Information Label.

4.3.3 Browser UI Templates

Figure 4.50 depicts a sample Browser Lightbox processing screen without a white box.

Figure 4.50: Sample Browser Lightbox Processing Screen without White Box

Figure 4.51 depicts a sample Inline Browser Processing screen with a white box.

Figure 4.51: Sample Browser Lightbox Processing Screen with White Box

Chapter 5 EMV 3-D Secure Message Handling Requirements

5.1 General Message Handling

5.1.1 HTTP POST

[Req 186]

The body of the HTTP message shall contain the JSON message properly formatted utilising the JSON required UTF-8 character set as defined in RFC 7159, or JWE/JWS object format as defined in RFC 7516/RFC 7515. To maximise the message content, it is recommended to remove any whitespace (space, carriage return, line feed etc.) characters outside of quoted strings of the JSON data.

5.1.2 HTTP Header—Content Type

[Req 190]

The HTTP headers shall contain the Content-Type Header: application/JSON; and include charset of UTF-8 for the following messages:

- AReq/ARes
- RReq/RRes
- PReq/PRes
- OReq/ORes
- Error Message

[Req 191]

The Content-Type Header requirements for CReq/CRes are:

- For Browser-based CRes, the HTTP headers shall contain the Content-Type Header: text/html and include charset of UTF-8.

  For example, Content-Type: text/html~~:~~; charset = UTF-8.

New wording was added directly after Requirement 191.

The HTTP headers contain additional information for the support of 3-D Secure messaging:

[Req 468]

For the AReq, CReq, RReq, OReq or PReq messages, the 3DS component sending the message shall include its own Transaction ID using the X-Request-ID in the HTTP header, as defined in Table A.29.

[Req 469]

For the ARes, CRes, RRes, ORes or PRes messages, the 3DS component sending the message shall include its own Transaction ID using X-Response-ID and the sender Transaction ID received in the Request message (not in the HTTP header), using the X-Request-ID, as defined in Table A.29.

5.1.4 Protocol and Message Version Numbers

Requirement 194 was converted to a Note.

~~**[Req 194]**~~

Note: ~~A 3-D Secure~~ Protocol and Message Version Numbers ~~shall be~~ are in the format major.minor.patch (for example, 2.3.1.~~0~~).

[Req 195]

Any Message Version Number not indicated as active in ~~Table 1.5~~ *EMV Specification Bulletin 255* shall be returned as an error. The 3-D Secure component shall return an Error Message (as defined in Section A.9) with the applicable Error Component and an Error Code = 102.

[Req 320]

The Message Version Number~~s shall be validated to ensure that they are consistent across a 3-D Secure transaction. The~~ is set by the 3-D Secure component initiating the 3DS message and the 3-D Secure components shall validate that the Message Version Number remains unchanged throughout the 3-D Secure transaction. The 3-D Secure component that identifies a validation error shall return an Error Message with the applicable Error Component (as defined in Section A.9) and Error Code = 203.

~~For example, when the DS receives an RReq message, the DS will validate that the Message Version Number matches the AReq message.~~

[Req 311]

3DS components shall support all lower active protocol versions (Protocol Version Status set to Active in *EMV Specification Bulletin 255*). 3DS components shall support latest patch for their current version. 3-D Secure messages containing an active Message Version Number supported by the 3-D Secure component shall be processed according to the requirements of the specified protocol version (See ~~Table 1.5~~ *EMV Specification Bulletin 255*).

~~**Note: For all 3-D Secure transactions, the 3DS Server sets the Message Version Number that all components will utilise.**~~

5.1.5 Data Version Numbers

[Req 396]

The 3DS SDK shall ~~implement~~ support the latest Data Version of the 3DS SDK Device Information.

[Req 397]

The ACS shall ~~implement~~ support all active Data Versions of the 3DS SDK Device Information.

Note: Refer to *EMV® Specification Bulletin 255* and *EMV® 3-D Secure SDK—Device Information*.

5.1.6 Message Parsing

When receiving a 3-D Secure message, the recipient shall validate that the:

[Req 201]

The 3DS Server shall only accept the following messages: ARes, RReq, PRes, OReq or Error Message. Any other message types shall be treated as an error.

[Req 202]

The DS shall only accept the following messages: AReq, ARes, RReq, RRes, PReq, ORes or Error Message. Any other message types shall be treated as an error.

[Req 203]

The ACS shall only accept the following messages: AReq, CReq, RRes, OReq or Error Message. Any other message types shall be treated as an error.

[Req 430]

If the 3DS Server receives more than one RReq message during a transaction, then it shall return Error Code = 312.

[Req 431]

If the 3DS Server receives an RReq message and the Transaction Status does not = C or D or S in the corresponding ARes message, then it shall return Error Code = 313.

[Req 432]

If the DS receives more than one RReq message during a transaction, then it shall return Error Code = 312.

[Req 433]

If the DS receives an RReq message and the Transaction Status does not = C or D or S in the corresponding ARes message, then it shall return Error Code = 313.

5.1.7 Message Content Validation

The message validation criteria are based on the Message Type field and apply as follows:

[Req 210]

All 3-D Secure components shall ~~silently~~ ignore unrecognised non-critical extension name/value pairs (that is, any extension that does not have a criticality attribute with a value = true) and pass them.

New Requirement 434 was added directly after Requirement 210.

[Req 434]

If the value of a data element is in the range of "Reserved for EMVCo future use", all 3DS components shall return an Error Message (as defined in Section A.9) with the applicable Error Component and Error Code = 207.

Note: The error requirements and the use of Error Code = 207 for the values in the range of "Reserved for DS use" are defined by the DS.

5.2 Partial System Outages

In Requirement 215, a reference to Section A.5.5 was replaced with a reference to Section A.9.

5.5 Timeouts

5.5.1 Transaction Timeouts

The ACS shall:

[Req 221]

If the transaction reaches the 30-second timeout expiry and an Error Message has not been received from the DS for this transaction, send an RReq message to the DS ~~to be passed to the 3DS Server~~ with Transaction Status = N, Transaction Status Reason = 14 (Transaction timed out at the ACS), and Challenge Cancelation Indicator = 05 (Transaction timed out at the ACS—First CReq message not received). Clear the ephemeral key generated and stored for use in the CReq/CRes message exchange for the current transaction.

[Req 222]

Upon receiving a CReq message for a transaction that has timed-out, send an Error Message (as defined in Section A.9) with Error Component = A and Error Code = 402 to the 3DS SDK (for an App-based transaction) or 3DS Requestor (for a Browser-based transaction).

For App-based transactions, once a transaction has been established with the initial CReq/CRes message exchange between the ACS and the 3DS SDK, and when the ACS sends a CRes message to the 3DS SDK that requires an additional CReq message to continue or complete the Cardholder challenge (Challenge Completion Indicator = N), the ACS shall:

[Req 223]

Set a timeout value of 10 minutes (or 600 seconds) after successfully sending each CRes message to the 3DS SDK.

[Req 224]

If the timeout expires before receiving the next CReq message from the 3DS SDK, send an RReq message within 60 seconds to the DS to be passed to the 3DS Server with Transaction Status = N, Transaction Status Reason = 14 (Transaction timed out at the ACS), and Challenge Cancelation Indicator = 04 and then clear any ephemeral key generated and stored for use in the CReq/CRes message exchange for this transaction.

[Req 227]

If the timeout expires before Cardholder authentication can complete, send an RReq message within 60 seconds to the DS to be passed to the 3DS Server with the Transaction Status = N and Transaction Status Reason = 14.

This completes the challenge for the ACS. In a timeout situation, the 3DS Server proceeds as defined in Section 3.3, Step 18 for the RReq and RRes messages. At the end of Step

18, the 3DS Server notifies the 3DS Requestor of the timeout. The method used to notify the 3DS Requestor is outside the scope of this specification.

~~**[Req 343]**~~

~~The ACS sends a CRes message with a Transaction Status = N to the Notification URL received in the initial AReq message.~~

~~This completes the challenge.~~

When notified of the timeout, the 3DS Requestor shall:

[Req 344]

Close the challenge ~~window upon receiving the CRes message~~ iframe by refreshing the parent page and removing the HTML iframe.

For an SPC challenge (Transaction Status = S), the ACS shall:

[Req 452]

Set a timeout value of 10 minutes (or 600 seconds) after sending the first ARes message.

[Req 453]

If the timeout expires before the second AReq message is received, send an RReq message within 60 seconds to the DS, to be passed to the 3DS Server, with Transaction Status = N and Transaction Status Reason = 14.

This completes the SPC challenge for the ACS. In a timeout situation, the 3DS Server proceeds as defined in Section 3.3, Step 18 for the RReq and RRes messages. At the end of Step 18, the 3DS Server notifies the 3DS Requestor of the timeout. The method used to notify the 3DS Requestor is outside the scope of this specification.

When notified of the timeout, the 3DS Requestor takes the appropriate action and message to the Cardholder.

For an SPC challenge (Transaction Status = S), the 3DS Server shall:

[Req 454]

Set a timeout value of 9 minutes (or 540 seconds) after successfully providing the SPC data to the 3DS Requestor.

[Req 455]

If the timeout expires before receiving the Assertion Data from the 3DS Requestor, send the second AReq message to the ACS with SPC Incompletion Indicator = 03 and without the 3DS Requestor Authentication Information.

5.5.2 Read Timeouts

5.5.2.1 AReq/ARes Message Timeouts

The 3DS Server:

[Req 229]

Any failure to complete the initial TCP/IP connection and TLS handshake to the DS shall result in an immediate retry or the 3DS Server shall try an alternate DS (if available). Upon second failure, the 3DS Server shall send an error to the 3DS Requestor to complete the transaction and may send an Error Message with Error Component = S and Error Code = 405 to the DS.

New Requirement 424 was added directly after the Note following Requirement 233.

The DS:

[Req 424]

If all attempts to successfully connect to the ACS fail, then the DS may send an Error Message with Error Component = D and Error Code = 405 to the ACS.

[Req 235]

If the DS has not received the ARes message from the ACS before the read timeout expiry, the DS shall send an:

- Error Message (as defined in Section A.9) with Error Component = D and Error Code = 402 to the 3DS Server to complete the transaction, OR
- ARes message to the 3DS Server (as defined in Table B.2) with Transaction Status set to the appropriate response as defined by the specific DS, and then send an Error Message (as defined in Section A.9) with Error Component = D and Error Code = 402 to the ACS to complete the transaction.

The 3DS SDK:

[Req 236]

Any failure to complete the initial connection and TLS handshake to the ACS shall result in an immediate retry. Upon second failure, the 3DS SDK shall ~~send~~ report an error to the 3DS Requestor App to complete the transaction.

5.5.2.2 RReq/RRes Message Timeouts

The ACS:

[Req 242]

If the DS has not responded with the RRes message or an Error message before the ~~5-second~~ read timeout expiry, the ACS shall return to the DS an Error Message (as defined in Section ~~A.5.5~~A.9) with Error Component = A and Error Code = 402. The default timeout value is 5 seconds. However, a DS may specify a higher alternative value.

The DS:

[Req 243]

Any failure to complete the initial connection and TLS handshake to the 3DS Server shall result in an immediate retry. Upon second failure, the DS shall send an Error Message with Error Component = D and Error Code = 405 to the ACS to complete the transaction and

may send an Error Message with Error Component = D and Error Code = 405 to the 3DS Server and **end 3-D Secure processing**.

~~**Note: No further processing shall occur between the DS and 3DS Server.**~~

[Req 244]

~~The DS s~~Shall set a ~~3-second~~ timeout value from the time the TLS handshake has completed and the full RReq message is sent for processing to the 3DS Server URL. The default timeout value is 3 seconds. However, a DS may specify a higher alternative value.

Note: When setting the timeout values, the DS ensures that the timeout value in [Req 242] is greater than the timeout value in [Req 244].

[Req 245]

If the 3DS Server has not sent the RRes message before the 3-second (or DS alternative value) read timeout expiry, the DS shall send an Error Message (as defined in Section A.9) with Error Component = D and Error Code = 402 to the ACS to complete the transaction and may send an Error Message with Error Component = D and Error Code = 402 to the 3DS Server and **ends 3-D Secure processing**.

~~**Note: No further processing shall occur between the DS and 3DS Server.**~~

5.6 PReq/PRes Message Handling Requirements

The PReq/PRes messages are utilised by the 3DS Server to cache information about the Protocol Version Numbers(s) supported by available ACSs, the DS, and also any URL to be used for the 3DS Method call. The data will be organised by card range as configured by a DS. The information provided on the Protocol Version Number(s) supported by ACSs and the DS can be ~~utilised in the App-based, Browser-based and 3RI~~ used in all 3DS flows.

The 3DS Server has the option to receive the Card Range Data in the PRes message or, if optionally supported by the DS, to receive a URL to download a file containing the Card Range Data.

The 3DS Server formats a PReq message (as defined in Table B.6) and sends the request to the DS. If this is the first time that the cache is being loaded (or if the cache has been flushed and needs to be reloaded, or if the DS does not support partial cache updates), the Serial Number data element is not included in the request, which will result in the DS returning the entire list of participating card range information.

Otherwise, the 3DS Server ~~should include~~ includes the Serial Number from the most recently processed PRes message, which will result in the DS returning only the changes since the previous PRes message. The Serial Number is not used or provided when the DS and 3DS Server use the Card Range Data File download option.

The DS manages the Serial Number to ensure that the response to a PReq message for a particular Serial Number includes all updates posted since that Serial Number was issued. If the Serial Number provided in the PReq message is invalid (for example, ~~if~~ too old and can no longer be found), the response should be an Error Message (as defined in Section A.9) with ~~an~~ Error Code = 307.

If the PReq message does not include a Serial Number, the DS PRes message response or file ~~shall contain~~ contains all card range entries.

If the Serial Number has not changed, the DS ~~would~~ does not provide back the Card Range Data element but ~~would include~~ includes the Serial Number in the PRes message ~~(i.e. it should be absent). Card Range Data should not be in the PRes; however, Serial Number should be included~~.

The 3DS Server shall:

[Req 246]

~~3DS Servers shall make a call to each registered DS every 24 hours at a minimum, and once per hour at a maximum to refresh their cache, conditional on no errors found during PRes message processing.~~

Call each registered DS for:

- An update for all Card Range Data (Serial Number not provided) every 12 hours at maximum. If there is an error in the received Card Range Data, the 3DS Server calls each registered DS once per hour at maximum for a complete or partial update.
- A partial update providing the Serial Number once per hour at maximum.

[Req 425]

Request the Card Range Data under a compressed format by adding “Accept-Encoding: gzip” to their HTTP request.

[Req 456]

Support Range Requests, as defined by RFC 7233, if the Card Range Data Download Indicator is present in the PReq message.

Requirements 247, 248 and 249 follow with no wording change.

The DS shall:

[Req 428]

If the DS receives a request without the “Accept-Encoding: gzip” in the header of an 3DS Server HTTP request, not return an Error Message and return the data in an uncompressed format.

Requirement 303 was rewritten for clarity, with no substantive change, to read as follows:

[Req 303]

~~Receive and validate the PReq message, as defined in Table B.6:~~

- ~~If any data element present fails validation, the DS:~~
  - ~~Returns to the 3DS Server an Error Message (as defined in Section A.5.5) with Error Component = D and Error Code = 203.~~
- ~~If any required data elements are missing, the DS:~~
  - ~~Returns to the 3DS Server an Error Message (as defined in Section A.5.5) with Error Component = D and Error Code = 201.~~
- ~~If the Serial Number is invalid, the DS:~~

- ~~Returns to the 3DS Server an Error Message (as defined in Section A.5.5) with Error Component = D and Error Code = 307.~~

- ~~If the 3DS Server submits more than one request for Card Range Data within one hour, the DS:~~
  - ~~Returns to the 3DS Server an Error Message (as defined in Section A.5.5) with Error Component = D and Error Code = 103.~~

Receive and validate the PReq message, as defined in Table B.6:

- Return to the 3DS Server an Error Message (as defined in Section A.9) with Error Component = D; AND
  - Error Code = 203, if any data element present fails validation
  - Error Code = 201, if any required data element is missing
  - Error Code = 307, if the Serial Number is invalid
  - Error Code = 103, if the 3DS Server submits more than one request for Card Range Data within one hour.

[Req 457]

Support Range Requests, as defined by RFC 7233, for the Card Range Data File download.

[Req 458]

Ensure that there is no overlap or conflict in the card ranges contained in the Card Range Data (Start and End of the card range).

[Req 459]

Provide in the Card Range Data data element only information about card ranges that are participating in EMV 3-D Secure and are registered with the DS that is responding to the request.

[Req 460]

Prepare the Card Range Data File if supported by the DS and if the Card Range Data Download Indicator was present in the PReq message. The Card Range Data File contains the entire list of participating card ranges, e.g. the JSON object Card Range Data; {"cardRangeData": [...] }.

[Req 426]

Prepare the PRes message and send a response with either:

- A compressed response body and a Content-Encoding header that specifies that gzip encoding was used (“Content-Encoding: gzip “), OR
- An uncompressed response body. The DS shall use the uncompressed response format when it provides the Card Range Data File URL in the PRes message.

[Req 250]

Send the PRes message containing the DS card range information as defined in the Card Range Data data element defined in Table A.1 or the Card Range Data File URL, if supported.

- If the PReq message does not include a Serial Number, or if the DS does not support partial cache update, the DS PRes message response shall contain ~~all~~the entire list of participating card ranges in the Card Range Data using only Action Indicator = A.

Requirement 251 was deleted.

~~**[Req 251]**~~

~~Send the PRes message containing only information about card account ranges that are participating in EMV 3-D Secure and are registered with the DS that is responding to the request.~~

Requirement 304 remains unchanged (with references to Section A.5.5 replaced with references to Section A.9) and is directly followed by new verbiage:

The 3DS Server retrieves the Card Range Data data element from the PRes message or from the file downloaded from the Card Range Data File URL. For the file download, it is recommended that the 3DS Server and the DS support a compressed format ("Accept-Encoding: gzip" in the HTTP request).

The 3DS Server shall:

[Req 385]

Update the cache information for each Card Range Data according to the Action Indicator.

- ~~If the PRes message does not include a Serial Number, the 3DS Server:~~
  - ~~Replaces all existing Card Range Data for the DS.~~
- ~~If an error is identified in the Card Range Data, the 3DS Server:~~
  - ~~Resubmits the PReq message without the Serial Number.~~
- If there is an error in the Card Range Data, then:
  - For a Card Range Data overlap, return to the DS an Error Message (as defined in Section A.9) with Error Component = S and Error Code = 205.
  - For an Action Indicator error, return to the DS an Error Message (as defined in Section A.9) with Error Component = S an Error Code = 206.
  - Discard all updates contained in the Card Range Data, and use previously stored cache information or alternatively, ignore all existing cache information.
- If the PRes message does not include a Serial Number, replace all existing Card Range Data for the DS using Action Indicator = A for all card ranges returned (i.e., the Action Indicator is ignored in the PRes message).

Note: Because Card Range Data could be large (e.g., 200 MB), the 3DS Server needs to ensure that they are equipped to process a large PRes file and reference DS guidelines for timeout values.

5.7 App/SDK-based Message Handling

The 3DS SDK shall be developed adhering to the applicable ~~EMV 3-D Secure – 3DS SDK Specification~~ specification requirements and APIs.

The 3DS SDK has two key functions:

- Provide all data as specified in *EMV 3-D Secure ~~3DS SDK—Device Information Specification~~* to be sent through the 3DS Requestor Environment to the 3DS Server and on to the DS and ACS.

5.7.1 App-based CReq/CRes Message Handling

The 3DS SDK—initialised for the Challenge Flows by the 3DS Requestor App as defined in the applicable ~~EMV 3-D Secure—~~3DS SDK ~~Specification~~specification—generates the CReq message using ARes message data received from the 3DS Server through the 3DS Requestor Environment.

5.8 Browser-based Message Handling

5.8.1 3DS Method Handling

The inclusion of 3DS Method URL and card ~~account~~ ranges in a DS is optional for an ACS.

New paragraph following the Note after Requirement 255.

If the 3DS Method URL is present for the card range, the 3DS Method is invoked for every start of a 3-D Secure Browser authentication transaction unless a prior 3DS Method for the same card, device and Browser has been successfully invoked in the last 10 minutes, in which case the 3DS Requestor can optionally reuse the result from the previous 3DS Method call.

5.8.1.1 Recent Prior 3DS Method Call Does Not Exist

If no prior 3DS Method call has been invoked in the last 10 minutes for the same Cardholder Account Number on the same device and Browser or if a recent prior 3DS Method call is not utilised, then the requirements in this section apply.

The 3DS Requestor shall:

[Req 256]

For the card ~~account~~ ranges that contain a 3DS Method URL in the cached PRes message, ~~the 3DS Requestor shall~~ invoke the 3DS Method.

[Req 257]

~~The~~ Invoke the 3DS Method call ~~shall occur~~ in advance of the AReq message for the same authentication transaction being sent ~~to the ACS~~.

Note: The 3DS Requestor determines ~~the timing of~~ when to start the 3DS Method call to optimise the user experience. The 3DS Method call could be invoked as soon as the 3DS Requestor has an indication of the Cardholder's intended payment card to minimise latency.

[Req 258]

Obtain from the 3DS Server or load from local cache, the 3DS Method URL if it exists for the card range.

If the 3DS Method URL does not exist, the 3DS Requestor ~~will notify~~notifies the 3DS Server to set the 3DS Method Completion Indicator = U.

Note: The 3DS Server Transaction ID is included in both the 3DS Method and the subsequent AReq message for the same transaction as defined in *[Req 83]*. ~~Refer to [Req 82] and [Req 84]~~.

Requirements 259, 260, 261, 262 and 263 follow with no substantive change.

The 3DS Server shall:

[Req 315]

Set the 3DS Method Completion Indicator = Y upon notification from the 3DS Requestor. If the 3DS Method does not complete within ~~10~~ 5 seconds, set the 3DS Method Completion Indicator to = N.

5.8.1.2 Recent Prior 3DS Method Call Does Exist

If a prior 3DS Method call has been invoked in the last 10 minutes, then the requirements in this section apply.

[Req 415]

If the 3DS Requestor has already processed a 3DS Method call with the same Cardholder Account Number on the same device and Browser in the last 10 minutes, then the 3DS Requestor may use the previous 3DS Method execution and choose not to invoke a fresh new 3DS Method call.

If a prior 3DS Method call is utilised, then the 3DS Server shall set the 3DS Method ID to the 3DS Server Transaction ID from the previous transaction and the Method Completion Indicator = Y in the AReq message.

5.9 Message Error Handling

Updated/renumbered references in this section are not reflected in this specification bulletin.

5.9.5 ACS CReq Message Error Handling—01-APP

The ACS processes the validation of the CReq message or Error Message as follows:

- For a correctly verified, decrypted, and recognised CReq message, the ACS Validates the CReq message as defined in Table B.3 and Section 5.1.7 according to the Device Channel = 01-APP:
  - If any required data elements are missing, the ACS:
    - Returns to the 3DS SDK an Error Message (as defined in Section A.9) with Error Component = A and Error Code = 201 using the secure link established in ***[Req 44]***.
    - If a specific transaction can be identified, sends to the DS an RReq message as defined in Section 5.9.5.1.
  - If SDK Type = 02 (as received in the AReq message) AND the CReq is not protected using A128GCM ("enc" as defined in Section 6.2.4.3 is not A128GCM), the ACS:

- Returns to the 3DS SDK an Error Message (as defined in Section A.9) with Error Component = A and Error Code = 310 using the secure link established in ***[Req 44]***.
- If a specific transaction can be identified, sends to the DS an RReq message as defined in Section 5.9.5.1.

- For an Error Message, if a specific transaction can be identified, the ACS ~~sends to the DS an RReq message as defined in Section 5.9.5.1~~.
  - Establishes a secure link with the DS as defined in Section 6.1.3.2.
  - Sends to the DS an RReq message (as defined in Table B.6) with Transaction Status = U and Challenge Cancelation Indicator = 09 using the secure link.

5.9.5.1 Message in Error

The ACS:

- Sends to the DS an RReq message (as defined in Table B.8) with Transaction Status = U and Challenge Cancelation Indicator = ~~06~~10 using the secure link.

5.9.6 ACS CReq Message Error Handling—02-BRW

The ACS processes the validation of the CReq message as follows:

- For a message that cannot be recognized, the ACS~~[7]~~: ~~– *Footnote:* Note that, for an un-recognised message, the ACS cannot extract the Notification URL used to communicate with the 3DS Server (via the Browser), thus a response is not possible.~~
- For multiple received CReq messages, the ACS:
  - Validates that the data elements from the CReq message are identical to the first CReq message.
  - If any data element present is different, the ACS:
    - Sends to the DS an RReq message (as defined in Section 5.9.5.1).
    - Returns (via the Browser) an Error Message (as defined in Section A.5.5) with Error Component = A and Error Code = 305.
  - If the subsequent CReq message is identical to the first CReq message, but the ACS does not proceed with the challenge, the ACS:
    - Sends to the DS an RReq message (as defined in Section 5.9.5.1).
    - Returns an Error message to the Notification URL (via the Browser) with Error Component = A and Error Code = 314.
  - If the subsequent CReq message is received after the ACS has sent the RReq message, the ACS:
    - Returns an Error message to the Notification URL (via the Browser) with Error Component = A and Error Code = 315.
- For a CReq message, the ACS Validates the CReq message (as defined in Table B.3 and Section 5.1.7) according to the Device Channel = 02-BRW:
  - If any data element present fails validation, the ACS:

- If the Notification URL is valid, returns (via the ~~3DS Client~~ Browser) an Error Message (as defined in Section A.9) and Error Component = A and Error Code = 203.

- If any required data elements are missing, the ACS:
  - If the Notification URL is present, returns (via the ~~3DS Client~~ Browser) an Error Message (as defined in Section A.9) with Error Component = A and Error Code = 201.
  - If a specific transaction can be identified, sends to the DS an RReq message (as defined in Section 5.9.5.1).
- Otherwise, the message is not in error.

5.9.8 DS RReq Message Error Handling

5.9.8.1 Message in Error

The DS:

- Establishes a secure link with the 3DS Server (as defined in Section 6.1.2.2) using the 3DS Server URL extracted from the AReq message and stored in:
  - [Req 22] for an App-based transaction OR
  - [Req 99] for a Browser-based transaction OR
  - [Req 427] for a 3RI-based transaction

5.9.10 DS RRes Message Error Handling

The DS processes the validation of the RRes message or Error Message as follows:

- For a message that cannot be recognised, the DS:
  - If a specific transaction can be identified, sends to the ACS an Error Message (as defined in Section A.9) with Error Component = D and Error Code = 101 using the secure link established in ***[Req 63]*** for an app-based transaction or ***[Req 125]*** for a Browser-based transaction, or ***[Req 350]*** for a 3RI transaction.
- For an RRes message, the DS Validates the RRes message (as defined in Table B.9 and Section 5.1.7):
  - If any data element present fails validation, the DS:
    - If a specific transaction can be identified, sends to the ACS an Error Message (as defined in Section A.9) with Error Component = D and Error Code = 203 using the secure link established in ***[Req 63]*** for an app-based transaction or ***[Req 125]*** for a Browser-based transaction, or ***[Req 350]*** for a 3RI transaction.
  - If any required data elements are missing, the DS:

- If a specific transaction can be identified, sends to the ACS an Error Message (as defined in Section A.9) with Error Component = D and Error Code = 201 using the secure link established in ***[Req 63]*** for an app-based transaction or ***[Req 125]*** for a Browser-based transaction, or ***[Req 350]*** for a 3RI transaction.

- For an Error message, if a specific transaction can be identified, the DS sends to the ACS the Error Message as received from the 3DS Server using the secure link established in ***[Req 63]*** for an app-based transaction or ***[Req 125]*** for a Browser-based transaction, or ***[Req 350]*** for a 3RI transaction.

5.9.13 ACS RRes Message Error Handling—03-3RI

The ACS processes the validation of the RRes message or Error Message as follows:

- For a message that cannot be recognised, the ACS:
  - Returns to the DS an Error Message (as defined in Section A.9) with Error Component = A and Error Code = 101.
- For an RRes message, the ACS Validates the RRes message (as defined in Table B.9 and Section 5.1.7):
  - If any data element present fails validation, the ACS:
    - Returns to the DS an Error Message (as defined in Section A.9) with Error Component = A and Error Code = 203.
  - If any required data elements are missing, the ACS:
    - Returns to the DS an Error Message (as defined in Section A.9) with Error Component = A and Error Code = 201.
  - Otherwise, the message is not in error.

5.10 UTC Date and Time

This section provides requirements for the handling of UTC date and time by the 3DS components.

[Req 416]

The maximum desynchronisation with UTC time for the 3DS Server, DS and ACS shall be less than 15 minutes when providing or verifying the UTC date and time data elements.

The DS shall respond to the 3DS Server with an Error Message with Error Component = D and Error Code = 203 if the SDK Signature Timestamp or Purchase Date & Time data elements are received with a UTC time difference greater than 30 minutes.

[Req 417]

The ACS may respond to the DS with an Error Message with Error Component = A and Error Code = 203 if the SDK Signature Timestamp or Purchase Date & Time data elements are received with a UTC time difference greater than 30 minutes.

Note: For UTC date and time related elements, the time is converted from local time to UTC. For example, the Purchase Date & Time of an authentication initiated in local time: 30 October 2020 at 15:45:26 (UTC-4) = 20201030194526 when converted into UTC.

5.11 OReq/ORes Message Handling Requirements

Operation Messages provide the DS with the ability to communicate operational information to a 3DS Server or to an ACS. Operation Messages can be independent of, or related to, a payment/non-payment authentication transaction.

Operation Messages can be used to convey operational information about the overall EMV 3DS program system health and management. For example:

- Reporting on turnaround times and performance
- Detecting and flagging rogue players in the ecosystem
- DS can communicate key exchange/certificate updates/reminders
- Exchanging information on compromised devices

The DS using Operation Message shall:

[Req 435]

Prepare and send the OReq message or the sequence of OReq messages in the sequence number order via a secure link with the recipient (3DS Server or ACS) established as defined in Table B.10.

[Req 436]

Immediately retry a connection upon any failure to complete the initial TCP/IP connection and TLS handshake to the recipient.

[Req 437]

Upon the second failure to complete the TCP/IP connection and TLS handshake to the recipient, end the transaction, and periodically retry within the 24-hour window at 60-second intervals until the transaction completes successfully.

The OReq Recipient (3DS Server or ACS) shall:

[Req 438]

Receive and validate all the OReq messages in the sequence as defined in Table B.10:

- If any data element present fails validation, the OReq recipient:
  - Returns to the DS an Error Message (as defined in Section A.9) with Error Component = S if the OReq recipient is the 3DS Server or Error Component = A if the OReq recipient is the ACS and Error Code = 203.
- If any required data elements are missing, the OReq recipient:
  - Returns to the DS an Error Message (as defined in Section A.9) with Error Component = S if the OReq recipient is the 3DS Server or Error Component = A if the OReq recipient is the ACS and Error Code = 201.

[Req 439]

Prepare and send the ORes message to the DS via the secure link as defined in Table B.11.

The DS using Operation Message shall:

[Req 440]

- Receive and validate the ORes message as defined in Table B.11:
- If any data element present fails validation, the DS:
  - Returns to the ORes sender (3DS Server or ACS) an Error Message (as defined in Section A.9) with Error Component = D and Error Code = 203.
- If any required data elements are missing, the DS:
  - Returns to the ORes sender (3DS Server or ACS) an Error Message (as defined in Section A.9) with Error Component = D and Error Code = 201.

Note: In case of a sequence of OReq messages, the DS sends all the OReq messages and the ACS responds with a single ORes message after receiving all the OReq messages.

Note: 3-D Secure processing completes.

Chapter 6 EMV 3-D Secure Security Requirements

6.1 Link

6.1.1 Link a: Consumer Device—3DS Requestor

The following sentence was added at the end of the section:

It is recommended to use TLS 1.2 or higher for all these links.

6.1.4.1 For App-based CReq/CRes

- Protocol—TLS ~~Internet~~1.2 or higher

6.1.4.2 For Browser-based CReq/CRes

- Protocol—TLS ~~Internet~~1.2 or higher

6.1.8 Link h: Browser—ACS (for 3DS Method)

- Protocol—TLS ~~Internet~~1.2 or higher

6.2 Security Functions

6.2.1 Function H: Authenticity of the 3DS SDK

3DS Requestors that deploy an EMVCo-approved 3DS SDK embedded in their App ~~and~~ are required to have a mechanism to authenticate the 3DS Requestor App to the 3DS Requestor, including confirmation that the embedded 3DS SDK has not been changed. For a 3DS Split-SDK this may be addressed via the intrinsic relationship between the Split-SDK Server and the Split-SDK Client components.

6.2.2 Function I: 3DS SDK Device Information Encryption and Split-SDK Server Signature to DS

This 3DS SDK to DS function occurs via the 3DS Server. The purpose is to allow ~~the~~:

- the 3DS SDK to encrypt data (e.g. Device Information) destined for the ACS. The decryption occurs at the DS that is trusted by the ACS, which consequently means that the data from the 3DS SDK is securely delivered to the ACS.
- a Split-SDK to provide a Split SDK-Server signature for confirmation by the DS.

6.2.2.1 3DS SDK Device Information Encryption

As a prerequisite, the SDK is loaded with (or has access to) the public key P_{DS} for the DS. There may be multiple keys for each DS with the keys each having an individual identifier PDSid. A UUID format is recommended for the PDSid.

The 3DS SDK:

- If P_{DS} is an RSA key:
  - Encrypts the JSON object according to JWE (RFC 7516) using JWE Compact Serialization. The parameter values for this version of the specification and to be included in the JWE protected header are:

- "alg": "RSA-OAEP-256"
- "kid":PDSid
- "enc": "A128CBC-HS256 or A128GCM"
- All other parameters ~~not present~~optional

- Else if P_{DS} is an ECC key:
  - Encrypts the JSON object according to JWE (RFC 7516) using the CEK and JWE Compact Serialization. The parameter values for this version of the specification and to be included in the JWE protected header are:
    - "alg":"ECDH-ES"
    - "kid":PDSid
    - "epk": Q_{SDK},
      {"kty": "EC",
      "crv": "P-256~~"~~}"
      "x":x coordinate of Q_{SDK}
      "y":y coordinate of Q_{SDK}}
    - "enc":either "A128CBC-HS256" or "A128GCM"
    - All other parameters: ~~not present~~optional

6.2.2.2 DS Device Information Decryption

The DS:

- If the protected header of the JWE in the SDK Encrypted Data field indicates that RSA-OAEP-256 was used for encryption:
  - Decrypts the SDK Encrypted Data field from the AReq message according to JWE (RFC 7516) using the parameter values from the protected header (RSA-OAEP-256 and P_{DS} as indicated by "kid") either A128CBC-HS256 or A128GCM as indicated by the "enc" parameter in the protected header.
- Else, if the protected header of the JWE in the SDK Encrypted Data field indicates that ECDH-ES was used for encryption:
  - ~~Decrypts the SDK Encrypted Data field as follows:~~
  - Conducts a Diffie-Hellman key exchange process according to JWA (RFC 7518) in Direct Key Agreement mode using the parameter values from the protected header (ECDH-ES, curve P-256, $Q_{SDK,}$ and d_{DS} by "kid") ~~with the parameter values from the protected header~~ and Concat KDF to produce a 256-bit CEK. The Concat KDF parameter values for this version of the specification are:
    - Keydatalen = 256
    - AlgorithmID = empty string (length = 0x00000000)
    - PartyUInfo = empty string (length = 0x00000000)
    - PartyVInfo = directoryServerID (length || ascii string)
    - SuppPubInfo = Keydatalen (0x00000100)
    - SuppPrivInfo = empty octet sequence

6.2.2.3 Split-SDK Server Signature

A Split-SDK Server creates a time-stamped signature on certain transaction data that is sent to the DS for verification.

As a prerequisite, the Split-SDK Server has a key pair (Pb_{SDK}, Pv_{SDK}) certificate Cert (Pb_{SDK}). This certificate is an X.509 certificate signed by a DS CA whose public key is known to the DS.

The Split-SDK Server:

- Creates a JSON object of the following data as the JWS payload to be signed:
  - SDK Reference Number
  - SDK Transaction ID
  - Split-SDK Server ID
  - SDK Signature Timestamp
- Generates a digital signature of the full JSON object according to JWS (RFC 7515) using JWS Compact Serialization. The parameter values for this version of the specification and to be included in the JWS header are:
  - "alg": PS256[7] ~~8F~~ or ES256 – *Footnote:* [7] PS256 (RSA-PSS) is specified in preference to RS256 (RSASSA-PKCS1-v1_5) following the recommendation in RFC 3447 (2003).
  - "x5c": X.5C v3: Cert (Pb_{SDK}) and chaining certificates, if present (without any DS CA public key certificate)
- All other parameters: optional
- Includes the resulting JWS in the AReq message as SDK Server Signed Content

6.2.2.4 DS Verification of Split-SDK Server Signed Content

The DS:

Using the CA public key of the DS CA, validates the JWS from the Split-SDK Server according to JWS (RFC7515) using either PS256 or ES256 as indicated by the "alg" parameter in the header. If validation fails, ceases processing and reports error.

If the Split-SDK Server Signature is valid, the DS has confirmed the authenticity of the Split-SDK Server and that the SDK Reference Number and date are correct.

6.2.3 Function J: 3DS SDK—ACS Secure Channel Set-Up

6.2.3.2 ACS Secure Channel Setup

The ACS:

- Generates a digital signature of the full JSON object according to JWS (RFC 7515) using JWS Compact Serialization. The parameter values for this version of the specification and to be included in the JWS header are:
  - "alg": PS256 or ES256
  - "x5c": X.5C v3: Cert(Pb_{ACS}) and chaining certificates, if present (without any DS CA public key certificate)

- All other parameters: ~~not present~~optional

- Zeros the channel counters ACSCounterAtoS (~~:octet~~ 8 bits) and ACSCounterStoA (~~:octet~~ 8 bits)

6.2.3.3 3DS SDK Secure Channel Setup

The 3DS SDK:

- Verifies that the SDK Qc present in the signature, is the same as generated by the SDK in Section 6.2.3.1 and provided for inclusion in the AReq message as sdkEphemPubKey.
- Completes the ~~Diffie-Hellman~~ DH key exchange process according to JWA (RFC 7518) in Direct Key Agreement mode using curve P-256, d_C and Q_T, with Concat KDF to produce a 256-bit CEK, which is identified to the ACS Transaction ID received in the ARes message. The Concat KDF parameter values ~~supported~~ for this version of the specification are:
- Zeros the channel counters SDKCounterAtoS (~~:octet~~ 8 bits) and SDKCounterStoA (~~:octet~~ 8 bits)

6.2.4 Function K: 3DS SDK—ACS (CReq/CRes)

6.2.4.1 3DS SDK—CReq

For CReq messages sent from the 3DS SDK to the ACS, the 3DS SDK:

- Encrypts the JSON object according to JWE (RFC 7516) using the CEK_{S-A} obtained in Section 6.2.3.3 and JWE Compact Serialization. The parameter values for this version of the specification and to be included in the JWE protected header are:
  - "alg": dir
  - "enc":
    - For SDK Type = 01: either: A128CBC-HS256 or A128GCM
    - For SDK Type = 02: A128GCM
  - "kid":ACS Transaction ID
  - All other parameters: ~~not present~~optional
- ~~Sends the resulting JWE to the ACS as the protected CReq message.~~
- Increments SDKCounterStoA.
  - If SDKCounterStoA ≠ zero, sends the resulting JWE to the ACS as the protected CReq message.
  - If SDKCounterStoA = zero, ceases processing and reports error.

6.2.4.2 3DS SDK—CRes

For CRes messages received by the 3DS SDK from the ACS, the 3DS SDK:

- Checks that ACSCounterAtoS in the decrypted message numerically equals SDKCounterAtoS. If not ceases processing and reports error.

6.2.4.4 ACS—CRes

For CRes messages sent from the ACS to the 3DS SDK the ACS:

- Encrypts the JSON object according to JWE (RFC 7516) using the same "enc" algorithm used by the 3DS SDK for the CReq message, the CEK_{A-S} obtained in Section 6.2.3.2 identified by "kid" and JWE Compact Serialization. The parameter values for this version of the specification and to be included in the JWE protected header are:
  - "alg": dir
  - "enc": either A128CBC-HS256 or A128GCM
  - "kid": ACS Transaction ID
  - All other parameters: ~~not present~~optional
- ~~Sends the resulting JWE to the 3DS SDK as the protected CRes message.~~
- Increments ACSCounterAtoS.
  - If ACSCounterAtoS ≠ zero, sends the resulting JWE to the 3DS SDK as the protected CReq message.
  - If ACSCounterAtoS = zero, ceases processing and reports error.

Annex A 3-D Secure Data Elements

Some sections of Annex A were moved within the annex and new sections were added. Please be advised that heading and table numbering has been updated since previous versions of the specification, and some discrepancies in table number references may exist. Some rows were rearranged in alphabetical order, without any effect on the meaning, and are therefore not replicated in this specification bulletin.

- Length/Format/Values—Identifies the value length detail, JSON data format, and if applicable, the values associated with the data element. The term "character" in the Length Edit criteria refers to one UTF-8 character. The length of a JSON object is the length in characters of the overall string representing the object.

A.4 EMV 3-D Secure Data Elements

Table A.1 EMV 3-D Secure Data Elements

Data Element/ Field Name	Description	Source	Length/Format/ Values	Device Channel	Message Category	Message Inclusion	Condition Inclusion
3DS Method Completion Indicator			N = Did not run or did not successfully complete				
3DS Method ID Field Name: `threeDSMethodId`	Contains the 3DS Server Transaction ID used during the previous execution of the 3DS method.	3DS Server	Length: 36 characters JSON Data Type: String Value accepted: Canonical format as defined in IETF RFC 4122. May utilise any of the specified versions if the output meets specified requirements.	02-BRW	01-PA 02-NPA	AReq = C	Required if 3DS Requestor reuses previous 3DS Method execution.

Data Element/ Field Name	Description	Source	Length/Format/ Values	Device Channel	Message Category	Message Inclusion	Condition Inclusion
3DS Requestor App URL	3DS Requestor App declaring its URL within the CReq message so that the Authentication App can call the 3DS Requestor App after OOB authentication has occurred. ~~Each transaction would require a unique Transaction ID by using the SDK Transaction ID.~~		Length: Variable, maximum ~~256~~ 2048 characters Value accepted: • ~~Fully Qualified URL~~ ~~OR~~ • Universal App Link ~~**Note: It is recommended to use Universal App Link.**~~				
3DS Requestor App URL Indicator Field Name: `threeDSRequestorAppURLInd`	Indicates whether the OOB Authentication App used by the ACS during a challenge supports the 3DS Requestor App URL.	ACS	Length: 1 character JSON Data Type: String Values accepted: • Y = 3DS Requestor App URL is supported by the OOB Authentication App • N = 3DS Requestor App URL is NOT supported by the OOB Authentication App	01-APP	01-PA 02-NPA	ARes = R	

Data Element/ Field Name	Description	Source	Length/Format/ Values	Device Channel	Message Category	Message Inclusion	Condition Inclusion
3DS Requestor Authentication Indicator		DS	Values accepted: • 08 = Split shipment • 09 = Delayed shipment • 10 = Split payment • ~~08~~11–79 = Reserved for EMVCo future use (values invalid until defined by EMVCo)				
3DS Requestor Authentication Information			Length: ~~Variable~~1–3 elements JSON Data Type: Array of objects Refer to ~~Table A.10~~Table A.12 for data elements to include. Note: Data will be formatted into a JSON Array of objects prior to being placed into the 3DS Requestor Authentication Information field of the message.				

Data Element/ Field Name	Description	Source	Length/Format/ Values	Device Channel	Message Category	Message Inclusion	Condition Inclusion
~~3DS Requestor Authentication Method Verification Indicator~~ ~~Field Name:~~ ~~threeDSReqAuthMethodInd~~	~~Value that represents the signature verification performed by the DS on the mechanism (e.g. FIDO) used by the Cardholder to authenticate to the 3DS Requestor.~~	~~DS~~	~~Length: 2 characters~~ ~~JSON Data Type: String~~ ~~Values accepted:~~ • ~~01 = Verified~~ • ~~02 = Failed~~ • ~~03 = Not performed~~ • ~~04–79 = Reserved for EMVCo future use (values invalid until defined by EMVCo)~~ • ~~80–99 = Reserved for DS use~~	~~01-APP~~ ~~02-BRW~~ ~~03-3RI~~	~~01-PA~~ ~~02-NPA~~	~~AReq = C~~	~~Conditional based on DS rules.~~ ~~The DS populates the AReq with this data element prior to passing to the ACS.~~
3DS Requestor Challenge Indicator	~~For local/regional mandates or other variables.~~ Note: When providing two preferences, the 3DS Requestor ensures that they are in preference order and not conflicting. For example, 02 = No challenge requested and 04 = Challenge requested (Mandate).	DS	Length: 1–2 ~~characters~~elements JSON Data Type: Array of string String: 2 characters Values accepted: • 08 = No challenge requested (utilise ~~whitelist~~ Trust List exemption if no challenge required)	03-3RI			

Data Element/ Field Name	Description	Source	Length/Format/ Values	Device Channel	Message Category	Message Inclusion	Condition Inclusion
			• 09 = Challenge requested (~~whitelist~~ Trust List prompt requested if challenge required) • 10 = No challenge requested (utilise low value exemption) • 11 = No challenge requested (Secure corporate payment exemption) • 12 = Challenge requested (Device Binding prompt requested if challenge required) • 13 = Challenge requested (Issuer requested) • 14 = Challenge requested (Merchant initiated transactions) • 15–79 = Reserved for EMVCo future use (values invalid until defined by EMVCo)				

Data Element/ Field Name	Description	Source	Length/Format/ Values	Device Channel	Message Category	Message Inclusion	Condition Inclusion
3DS Requestor Decoupled Max Time			Numeric values between 00001 and 10080 accepted.				Required if 3DS Requestor Decoupled Request Indicator = Y or F or B.
3DS Requestor Decoupled Request Indicator	Note: if the element is not provided, the expected action is for the ACS to interpret as N (Do not use Decoupled Authentication).		Values accepted: • Y = Decoupled Authentication is supported and is preferred as a primary challenge method if a challenge is necessary (Transaction Status = D in ARes). • N = Do not use Decoupled Authentication • F = Decoupled Authentication is supported and is to be used only as a fallback challenge method if a challenge is necessary (Transaction Status = D in RReq).				

Data Element/ Field Name	Description	Source	Length/Format/ Values	Device Channel	Message Category	Message Inclusion	Condition Inclusion
			• B = Decoupled Authentication is supported and can be used as a primary or fallback challenge method if a challenge is necessary (Transaction Status = D in either ARes or RReq). ~~Note: if the element is not provided, the expected action is for the ACS to interpret as N, do not use Decoupled Authentication.~~				
3DS Requestor ID	DS ~~assigned~~ defined 3DS Requestor identifier. ~~Each DS will provide a unique ID to each 3DS Requestor on an individual basis.~~		Values accepted: Any individual DS may impose specific formatting, ~~and~~ character and/or other requirements on the contents of this field.				

Data Element/ Field Name	Description	Source	Length/Format/ Values	Device Channel	Message Category	Message Inclusion	Condition Inclusion
3DS Requestor Name	DS ~~assigned~~defined 3DS Requestor name. ~~Each DS will provide a unique name to each 3DS Requestor on an individual basis.~~		Values accepted: Any individual DS may impose specific formatting, ~~and~~ character and/or other requirements on the contents of this field.				
3DS Requestor Prior Transaction Authentication Information			Length: Variable, 1–3 elements JSON Data Type: Array of objects Refer to Table ~~A.11~~A.13 for data elements to include.			AReq = ~~O~~C	~~Optional, recommended to include.~~ Required for 3RI in the case of Decoupled Authentication Fallback or for SPC
3DS Requestor SPC Support Field Name: `threeDSRequestorSpcSupport`	Indicate if the 3DS Requestor supports the SPC authentication. Note: If present, this field contains the value Y.	3DS Server	JSON Data Type: String Values accepted: • Y = Supported	02-BRW	01-PA 02-NPA	AReq = C	Required if supported by the 3DS Requestor
3DS Server Reference Number						ORes = C	Required in ORes message for a 3DS Server receiving an OReq message.

Data Element/ Field Name	Description	Source	Length/Format/ Values	Device Channel	Message Category	Message Inclusion	Condition Inclusion
3DS Server Transaction ID						ORes = C	• Required in ORes message for a 3DS Server receiving an OReq message.

Data Element/ Field Name	Description	Source	Length/Format/ Values	Device Channel	Message Category	Message Inclusion	Condition Inclusion
3RI Indicator			Values accepted: • 06 = Split~~/delayed~~ shipment • 10 = ~~Whitelist~~ Trust List status check • 13 = Device Binding status check • 14 = Card Security Code status check • 15 = Delayed shipment • 16 = Split payment • 17 = FIDO credential deletion • 18 = FIDO credential registration • 19 = Decoupled Authentication Fallback • 20–79 = Reserved for EMVCo future use (values invalid until defined by EMVCo)				

Data Element/ Field Name	Description	Source	Length/Format/ Values	Device Channel	Message Category	Message Inclusion	Condition Inclusion
Accept Language Field Name: acceptLanguage	Value representing the Browser language preference present in the HTTP header, as defined in IETF BCP 47.	3DS Server	Size: Variable, 1–99 elements JSON Data Type: Array of string String: Variable, maximum 100 characters	02-BRW	01-PA 02-NPA	AReq = R	
Acquirer BIN			• This value correlates to the Acquirer BIN as defined by each Payment System ~~or DS~~.				
Acquirer Country Code Field Name: acquirerCountryCode	The code of the country where the acquiring institution is located (in accordance with ISO 3166). The DS may edit the value provided by the 3DS Server.	3DS Server DS	Length: 3 characters JSON Data Type: String Values accepted: • ISO 3166-1 numeric three-digit country codes, other than exceptions listed in Table A.5.	01-APP 02-BRW 03-3RI	01-PA 02-NPA	AReq = R	
Acquirer Country Code Source Field Name: acquirerCountryCodeSource	This data element is populated by the system setting the Acquirer Country Code. The DS may edit the value provided by the 3DS Server.	3DS Server DS	Length: 2 characters JSON Data Type: String Value accepted: • 01 = 3DS Server • 02 = DS	01-APP 02-BRW 03-3RI	01-PA 02-NPA	AReq = R	

Data Element/ Field Name	Description	Source	Length/Format/ Values	Device Channel	Message Category	Message Inclusion	Condition Inclusion
			• 03–79 = Reserved for EMVCo future use (values invalid until defined by EMVCo) • 80–99 = Reserved for DS use				
Acquirer Merchant ID	This may be the same value that is used in authorisation requests sent on behalf of the 3DS Requestor and is represented in ISO 8583-1 formatting requirements.						
ACS Counter ACS to SDK	Note: The counter is the decimal value equivalent of the byte, encoded as a numeric string.		Values accepted: • 000–255				
ACS Ephemeral Public Key (Q_T)	~~The data element is contained within ACS Signed Content JWS Object.~~						

Data Element/ Field Name	Description	Source	Length/Format/ Values	Device Channel	Message Category	Message Inclusion	Condition Inclusion
ACS HTML			Length: Variable, maximum ~~100 kB~~300000 characters				~~Conditional upon selection of the ACS UI Type = 5 (HTML) by the ACS.~~ Required if ACS UI Type = 05 or 06.
ACS Reference Number						ORes = C	Required in ORes message for an ACS receiving an OReq message.
ACS Signed Content			Length: Variable, maximum 16000 characters				•
ACS Transaction ID						ORes = C	• Required in ORes message for an ACS receiving an OReq message.
ACS UI Type			Values accepted: • 06 = HTML OOB • 07 = Information			CRes = ~~R~~C	Required except for Final CRes message.

Data Element/ Field Name	Description	Source	Length/Format/ Values	Device Channel	Message Category	Message Inclusion	Condition Inclusion
			• ~~06~~08–79 = Reserved for EMVCo future use (values invalid until defined by EMVCo)				
App IP Address Field Name: `appIp`	External IP address (i.e., the device public IP address) used by the 3DS Requestor App when it connects to the 3DS Requestor Environment.	3DS Server	Length: Variable, maximum 45 characters JSON Data Type: String Value accepted: • IPv4 address. Refer to RFC 791. • IPv6 address. Refer to RFC 4291.	01-APP	01-PA 02-NPA	AReq = C	Required unless market or regional mandate restricts sending this information.

Data Element/ Field Name	Description	Source	Length/Format/ Values	Device Channel	Message Category	Message Inclusion	Condition Inclusion
Authentication Method	Indicates the list of authentication types the Issuer will use to challenge the Cardholder when in the ARes message or what was used by the ACS when in the RReq message. Note: For 03-3RI, only present for Decoupled Authentication. ~~Authentication approach that the ACS used to authenticate the Cardholder for this specific transaction.~~ ~~Note: This is in the RReq message from the ACS only. It is not passed to the 3DS Server.~~		~~Length: 2 characters~~Size: Variable, 1–99 elements JSON Data Type: Array of string String: 2 characters Values accepted: • 12 = Decoupled • 13 = WebAuthn • 14 = SPC • 15 = Behavioural biometrics • 16 = Electronic ID • ~~12~~17–79 = Reserved for EMVCo future use (values invalid until defined by EMVCo) If SDK Type and Split-SDK Type/Limited Indicator = Y, a value of 01 or 06 is not valid.			ARes = C	~~Required to be sent by the ACS.~~ ~~This field is present in the RReq message from the ACS to the DS but is not present in the RReq message from the DS to the 3DS Server.~~ • Required in the ARes message if Transaction Status = C or D. • Required in the RReq message if Transaction Status = Y or N.

Data Element/ Field Name	Description	Source	Length/Format/ Values	Device Channel	Message Category	Message Inclusion	Condition Inclusion
~~Authentication Type~~	~~Indicates the type of authentication method the Issuer will use to challenge the Cardholder, whether in the ARes message or what was used by the ACS when in the RReq message.~~	~~ACS~~	~~Length: 2 characters~~ ~~JSON Data Type: String~~ ~~Values accepted:~~ ~~01 = Static~~ ~~02 = Dynamic~~ ~~03 = OOB~~ ~~04 = Decoupled~~ ~~05–79 = Reserved for EMVCo future use (values invalid until defined by EMVCo)~~ ~~80–99 = Reserved for DS use~~	~~01-App~~ ~~02-BRW~~ ~~03-3RI~~	~~01-PA~~ ~~02-NPA~~	~~ARes = C~~ ~~RReq = C~~	~~Required in the ARes message if the Transaction Status = C or D in the ARes message.~~ ~~Required in the RReq message if the Transaction Status = Y or N in the RReq message.~~
Authentication Value			Length: ~~28 characters~~ Variable, maximum 4000 characters. Actual length defined by Payment System rules.				
Broadcast Information	~~Unstructured~~ Structured information sent between the 3DS Server, the DS and the ACS.		Refer to Table A.27 for data elements to include.			AReq = ~~C~~O ARes = ~~C~~O	~~Requirements for the presence of this field are DS specific.~~

Data Element/ Field Name	Description	Source	Length/Format/ Values	Device Channel	Message Category	Message Inclusion	Condition Inclusion
Browser IP Address			Values accepted: • IPv4 address ~~is represented in the dotted decimal format of 4 sets of decimal numbers separated by dots. The decimal number in each and every set is in the range 0 to 255~~. Refer to RFC 791. • IPv6 address ~~is represented as eight groups of four hexadecimal digits, each group representing 16 bits (two octets. The groups are separated by colons (:)~~. Refer to RFC 4291. ~~Refer to Section A.5.2 for additional detail~~				~~Shall include this field where regionally acceptable.~~ Required unless market or regional mandate restricts sending this information.
Browser Language			Length: Variable, ~~1–8~~ Maximum 35 characters			AReq = ~~R~~C	Required if Browser JavaScript Enabled = true; otherwise, Optional.

Data Element/ Field Name	Description	Source	Length/Format/ Values	Device Channel	Message Category	Message Inclusion	Condition Inclusion
Browser Screen Color Depth			Length: 1–2 characters; numeric JSON Data Type: String Values accepted: 1–99 Note: If an ACS does not support the provided value, then the ACS can use the closest supported value. For example, if the value provided = 30 and the ACS does not support that value, then the ACS could use the value = 24.				
Browser User Device ID Field Name: `deviceId`	Unique and immutable identifier linked to a device that is consistent across 3DS transactions for the specific user device. Examples: • Hardware Device ID • Platform-calculated device fingerprint Refer to D021 in the SDK Device Information	3DS Server	Length: Variable, maximum 64 characters JSON Data Type: String	02-BRW	01-PA 02-NPA	AReq = C	Required if available.

Data Element/ Field Name	Description	Source	Length/Format/ Values	Device Channel	Message Category	Message Inclusion	Condition Inclusion
Browser User ID Field Name: `userId`	Identifier of the transacting user's Browser Account ID. This identifier is a unique immutable hash of the user's account identifier for the given Browser, provided as a string. Note: Cardholders may have more than one account on a given Browser. Refer to D026 in the SDK Device Information	3DS Server	Length: Variable, maximum 64 characters JSON Data Type: String	02-BRW	01-PA 02-NPA	AReq = C	Required if available.
Card Range Data	Additionally, identifies the 3DS features the ACS supports, ~~for example, Whitelisting~~ such as Trust List or Decoupled Authentication.		~~Length~~Size: Variable, 1–200,000 elements JSON Data Type: Array of objects			PRes = ~~O~~C	Required if the Serial Number has changed in the prior PRes message or is absent in the PReq message AND Not present if the Card Range Data File URL is present

Data Element/ Field Name	Description	Source	Length/Format/ Values	Device Channel	Message Category	Message Inclusion	Condition Inclusion
Card Range Data Download Indicator Field Name: `cardRangeDataDownloadInd`	Indicates if the 3DS Server supports Card Range Data from a file. Note: If present, this field contains the value Y.	3DS Server	Length: 1 character JSON Data Type: String Value accepted: Y = Download supported	N/A	N/A	PReq = C	Present only if the 3DS Server supports the Card Range Data File download
Card Range Data File URL Field Name: `cardRangeDataFileURL`	Fully Qualified URL of the DS File containing the Card Range Data for download.	DS	Length: Variable, maximum 2048 characters JSON Data Type: String Value accepted: • Fully Qualified URL Example: https://server.dsdomainname.com/cardfile.json	N/A	N/A	PRes = C	Present only if the 3DS Server and the DS are using the Card Range Data File download

Data Element/ Field Name	Description	Source	Length/Format/ Values	Device Channel	Message Category	Message Inclusion	Condition Inclusion
Card Security Code Field Name: `cardSecurityCode`	Three- or four-digit security code printed on the card.	3DS Server	Length: Variable, 3-4 characters, numeric. Action defined by Payment System rules. JSON Data Type: String	01-APP 02-BRW 03-3RI	01-PA 02-NPA	AReq = C	Conditional based on DS rules
Card Security Code Status Field Name: `cardSecurityCodeStatus`	Enables the communication of Card Security Code Status between the ACS, the DS and the 3DS Requestor	ACS DS	Length: 1 character JSON Data Type: String Values accepted: • Y = Validated • N = Failed validation • U = Status unknown, unavailable, or does not apply	01-APP 02-BRW 03-3RI	01-PA 02-NPA	AReq = C ARes = C	Conditional based on DS rules
Card Security Code Status Source Field Name: `cardSecurityCodeStatusSource`	This data element will be populated by the system setting Card Security Code Status.	ACS DS	Length: 2 characters JSON Data Type: String Values accepted: • 01 = DS • 02 = ACS • 03–79 = Reserved for EMVCo future use (values invalid until defined by EMVCo) • 80–99 = Reserved for DS use	01-APP 02-BRW 03-3RI	01-PA 02-NPA	AReq = C ARes = C	Required if the Card Security Code Status is present.

Data Element/ Field Name	Description	Source	Length/Format/ Values	Device Channel	Message Category	Message Inclusion	Condition Inclusion
Cardholder Account Number	May be represented by PAN, Payment T~~t~~oken.						
Cardholder Billing Address State			~~Should be the~~Country subdivision code defined in ISO 3166-2. For example, using the ISO entry US-CA (California, United States), the correct value for this field = CA. Note that the country and hyphen are not included in this value.				

Cardholder Information Text	For example, “Additional authentication is needed for this transaction, please contact (Issuer Name) at xxx-xxx-xxxx” with optionally the Issuer and Payment System images.
	
	Refer to A.20 for UI example.
			Length: Variable, ~~maximum 128 characters~~
			
			JSON Data Type: ~~String~~Object
			
			Required:
			• `text`
			  ○ JSON Data Type: String
			  ○ Variable, 1–128 characters
			
			Optional:
			• `issuerImage`
			  ○ JSON Data Type: String
			  ○ Variable, maximum 256 characters
			  ○ Value accepted: Fully Qualified URL
			• `paymentSystemImage`
			  ○ JSON Data Type: String
			  ○ Variable, maximum 256 characters
			  ○ Value accepted: Fully Qualified URL
						RReq = O	

Data Element/ Field Name	Description	Source	Length/Format/ Values	Device Channel	Message Category	Message Inclusion	Condition Inclusion
			~~Note: If field is populated this information is required to be conveyed to the cardholder by the merchant.~~				
Cardholder Name			Length: Variable, ~~2~~1–45 characters ~~Value accepted:~~ ~~Alphanumeric special characters, listed in EMV Book 4, "Appendix B".~~				
Cardholder Shipping Address State			~~Should be The c~~Country subdivision code defined in ISO 3166-2. For example, using the ISO entry US-CA (California, United States), the correct value for this field = CA. Note that the country and hyphen are not included in this value.				

Data Element/ Field Name	Description	Source	Length/Format/ Values	Device Channel	Message Category	Message Inclusion	Condition Inclusion
Challenge Additional Code Field Name: challengeAddCode	Indicates to the ACS that the Cardholder selected the additional choice. Note: If present, this field contains the value Y.	3DS SDK	Length: 1 character JSON Data Type: String Values accepted: • Y = Additional choice selected	01-APP	01-PA 02-NPA	CReq = C	Required for Native UI: • if the Challenge Additional Label was present in the CRes message AND • if ~~the ACS offers~~ the Challenge ~~a~~Additional choice ~~button~~ is selected.
Challenge Additional Label Field Name: challengeAddLabel	UI label for the additional choice button provided by the ACS.	ACS	Length: Variable maximum 45 characters JSON Data Type: String	01-APP	01-PA 02-NPA	CRes = C	See Table A.20 for presence conditions.

Data Element/ Field Name	Description	Source	Length/Format/ Values	Device Channel	Message Category	Message Inclusion	Condition Inclusion
Challenge Cancelation Indicator			• 09 = Error message in response to the CRes message sent by the ACS • 10 = Error in response to the CReq message received by the ACS • 11–79 = Reserved for EMVCo future use				

Data Element/ Field Name	Description	Source	Length/Format/ Values	Device Channel	Message Category	Message Inclusion	Condition Inclusion
Challenge Data Entry	Note: ACS UI Type = 04, 05, 06 and 07 are not supported. Example: "challengeSelectInfo":[{"phon":"Mobile **** **** 321"}, {"mail":"Email a*******g**@g***.com"}] The Cardholder selects the phone option: "challengeDataEntry":"phon"						Required if: • ACS UI Type = 01, 02, or 03, AND • Challenge data has been entered in the Native UI text, AND • Challenge Cancelation Indicator is not present AND • Resend Challenge Information Code is not present AND • Challenge Additional Code is not present

Data Element/ Field Name	Description	Source	Length/Format/ Values	Device Channel	Message Category	Message Inclusion	Condition Inclusion
Challenge Data Entry 2 Field Name: challengeDataEntryTwo	Contains the data that the Cardholder entered into the Native UI text field. Note: Supported only for ACS UI Type = 01.	3DS SDK	Length: Variable, maximum 45 characters JSON Data Type: String	01-APP	01-PA 02-NPA	CReq = C	Required if: • ACS UI Type = 01, AND • Challenge Entry Box 2 object is provided by the ACS, AND • Challenge data has been entered in the second entry box/UI, AND • Challenge Cancelation Indicator is not present, AND • Resend Challenge Information Code is not present, AND • Challenge Additional Code is not present See Table A.16 for Challenge Data Entry conditions.

Data Element/ Field Name	Description	Source	Length/Format/ Values	Device Channel	Message Category	Message Inclusion	Condition Inclusion
Challenge Entry Box Field Name: `challengeEntryBox`	Defines the setting of an entry box in the Native UI OTP/Text Template: • Challenge Data Entry Keyboard Type • Challenge Data Entry Autofill • Challenge Data Entry Autofill Type • Challenge Data Entry Length Maximum • Challenge Data Entry Label • Challenge Data Entry Masking • Challenge Data Entry Masking Toggle	ACS	Length: Variable JSON Data Type: Object Values accepted: Refer to Table A.26	01-APP	01-PA 02-NPA	CRes = C	See Table A.20 for conditions.

Data Element/ Field Name	Description	Source	Length/Format/ Values	Device Channel	Message Category	Message Inclusion	Condition Inclusion
Challenge Entry Box 2 Field Name: challengeEntryBoxTwo	Defines the setting of an entry box in the Native UI OTP/Text Template: • Challenge Data Entry Keyboard Type • Challenge Data Entry Autofill • Challenge Data Entry Autofill Type • Challenge Data Entry Length Maximum • Challenge Data Entry Label • Challenge Data Entry Masking • Challenge Data Entry Masking Toggle	ACS	Length: Variable JSON Data Type: Object Values accepted: Refer to Table A.26	01-APP	01-PA 02-NPA	CRes = C	See Table A.20 for conditions.
Challenge Error Reporting Field Name: challengeErrorReporting	Copy of the Erro Message sent or received by the ACS in case of error in the CReq/CRes messages.	ACS	Length: Variable JSON Data Type: Object Values accepted: Refer to Table B.12 for data elements	01-APP 02-BRW	01-PA 02-NPA	RReq = C	Required if Challenge Cancelation Indicator = 09 or 10.
Challenge HTML Data Entry	Note: ACS UI Types 01, 02, 03, 04 and 07 are not supported.						Required if: • ACS UI Type = 05 or 06, AND

Data Element/ Field Name	Description	Source	Length/Format/ Values	Device Channel	Message Category	Message Inclusion	Condition Inclusion
							• Challenge Cancelation Indicator is not present, AND • OOB Continuation Indicator is NOT = 02
Challenge Information Header						CRes = ~~O~~C	See Table A.20 for presence conditions.
Challenge Information Label	~~Label to modify the Challenge Data Entry field~~ Text provided to the Cardholder by the ACS/Issuer to specify the expected challenge entry.					CRes = ~~O~~C	See Table A.20 for presence conditions.
Challenge Information Text			Note: Bold text is supported in this data element and is enclosed between **. Example: • "This is **bold** text" is rendered as "This is **bold** text".			CRes = ~~O~~C	See Table A.20 for presence conditions.

Data Element/ Field Name	Description	Source	Length/Format/ Values	Device Channel	Message Category	Message Inclusion	Condition Inclusion
Challenge Information Text Indicator			Length: 1 character			CRes = ~~O~~C	See Table A.20 for presence conditions.
Challenge No Entry							Required if: • Challenge Data Entry 2 is not present when Challenge Entry Box 2 object was provided by the ACS, AND • Challenge Additional Code is not present
Challenge Selection Information	Example: "challengeSelectInfo":[{"phon":"Mobile **** **** 321"}, {"mail":"Email a*******g**@g***.com"}]		Size: Variable, 1–8 elements ~~Length: Variable, each name/value pair maximum 45 characters~~ JSON Data Type: Array of objects Object: String key/value pair Key length: Variable, maximum 4 characters			CRes = ~~O~~C	See Table A.20 for presence conditions.

Data Element/ Field Name	Description	Source	Length/Format/ Values	Device Channel	Message Category	Message Inclusion	Condition Inclusion
			Value length: Variable, maximum 45 characters Note: If the field is populated this information is displayed to the Cardholder.				

Default-SDK Type
Field Name: `defaultSdkType`
	Indicates the characteristics of a Default-SDK.
	SDK Variant: SDK implementation characteristics
	Wrapped Indicator: If the Default-SDK is embedded as a wrapped component in the 3DS Requestor App
	Example:
	`"defaultSdkType":{`
	`"sdkVariant":"01",`
	`"wrappedInd":"Y"`
	`}`
		3DS Server	JSON Data Type: Object
			`sdkVariant`
			Length: 2 characters
			JSON Data Type: String
			Values accepted:
			• 01 = Native
			• 02–79 = Reserved for EMVCo future use (values invalid until defined by EMVCo)
			• 80–99 = Reserved for DS use
			`wrappedInd`
			Length: 1 character
			JSON Data Type: String
			Value accepted:
			• Y = Wrapped
			Only present if value = Y
				01-APP	01-PA
					02-NPA
						AReq = C	Required if SDK Type = 01

Data Element/ Field Name	Description	Source	Length/Format/ Values	Device Channel	Message Category	Message Inclusion	Condition Inclusion
Device Binding Data Entry Field Name: deviceBindingDataEntry	Indicator provided by the 3DS SDK to the ACS to confirm whether the Cardholder gives consent to bind the device.	3DS SDK	Length: 1 character JSON Data Type: String Values accepted: • Y = Consent given to bind device • N = Consent not given to bind device	01-APP	01-PA 02-NPA	CReq = C	Required if: • Device Binding Information Text was present in the previous CRes message AND • Challenge Cancelation Indicator is not present.
Device Binding Information Text Field Name: deviceBindingInfoText	Text provided by the ACS/Issuer to Cardholder during a Device Binding transaction. Example: • "Would you like to be remembered on this device?"	ACS	Length: Variable, maximum 64 characters	01-APP	01-PA 02-NPA	CRes = C	See Table A.20 for presence conditions.
Device Binding Status Field Name: deviceBindingStatus	Enables the communication of Device Binding Status between the ACS, the DS and the 3DS Requestor.	3DS Server DS ACS	Length: 2 characters JSON Data Type: String Values accepted: • 01 = Device is not bound by Cardholder • 02 = Not eligible as determined by Issuer	01-APP 02-BRW 03-3RI	01-PA 02-NPA	AReq = O ARes = O RReq = O	

Data Element/ Field Name	Description	Source	Length/Format/ Values	Device Channel	Message Category	Message Inclusion	Condition Inclusion
	For bound devices (value = 11–14), Device Binding Status also conveys the type of binding that was performed.		• 03 = Pending confirmation by Cardholder • 04 = Cardholder rejected • 05 = Device Binding Status unknown, unavailable, or does not apply • 06–10 = Reserved for EMVCo future use (values invalid until defined by EMVCo) • 11 = Device is bound by Cardholder (device is bound using hardware / SIM internal to the Consumer Device. For instance, keys stored in a secure element on the device) • 12 = Device is bound by Cardholder (device is bound using hardware external to the Consumer Device. For example, an external FIDO Authenticator				

Data Element/ Field Name	Description	Source	Length/Format/ Values	Device Channel	Message Category	Message Inclusion	Condition Inclusion
			• 13 = Device is bound by Cardholder (Device is bound using data that includes dynamically generated data and could include a unique device ID) • 14 = Device is bound by Cardholder (Device is bound using static device data that has been obtained from the Consumer Device • 15 = Device is bound by Cardholder (Other method) • 16–79 = Reserved for EMVCo future use (values invalid until defined by EMVCo) • 80–99 = Reserved for DS use				

Data Element/ Field Name	Description	Source	Length/Format/ Values	Device Channel	Message Category	Message Inclusion	Condition Inclusion
Device Binding Status Source Field Name: `deviceBindingStatusSource`	This data element will be populated by the system setting Device Binding Status.	3DS Server DS ACS	Length: 2 characters JSON Data Type: String Values accepted: • 01 = 3DS Server • 02 = DS • 03 = ACS • 04-79 = Reserved for EMVCo future use (values invalid until defined by EMVCo) • 80-99 = Reserved for DS use	01-APP 02-BRW 03-3RI	01-PA 02-NPA	AReq = C ARes = C RReq = C	Required if Device Binding Status is present.
Device Information			Refer to *EMV 3-D Secure SDK ~~Specification~~ Device Information* for values.				

Data Element/ Field Name	Description	Source	Length/Format/ Values	Device Channel	Message Category	Message Inclusion	Condition Inclusion
Device Information Recognised Version Field Name: deviceInfoRecognisedVersion	Indicates the highest Data Version of the Device Information supported by the ACS.	ACS	Length: Variable, minimum 3 characters JSON Data Type: String Values accepted: • Any active Device Information Data Version is considered a valid value. Refer to *EMV® Specification Bulletin 255* for values.	01-APP	01-PA 02-NPA	ARes = R	
~~DS End Protocol Version~~ ~~Field Name: dsEndProtocolVersions~~	~~The most recent active protocol version that is supported for the DS.~~ ~~Note: Optional within the Card Range Data (as defined in Table A.6).~~	~~DS~~	~~Length: Variable, 5–8 characters~~ ~~JSON Data Type: String~~	~~N/A~~	~~N/A~~	~~PRes = R~~	
~~DS Start Protocol Version~~ ~~Field Name: dsStartProtocolVersion~~	~~The earliest (i.e. oldest) active protocol version that is supported for the DS.~~ ~~Optional within the Card Range Data (as defined in Table A.6).~~	~~DS~~	~~Length: Variable, 5–8 characters~~ ~~JSON Data Type: String~~	~~N/A~~	~~N/A~~	~~PRes = R~~	

Data Element/ Field Name	Description	Source	Length/Format/ Values	Device Channel	Message Category	Message Inclusion	Condition Inclusion
DS Protocol Versions Field Name: `dsProtocolVersions`	Contains the list of active protocol versions supported by the DS. Note: Optional within the Card Range Data (as defined in Table A.6).	DS	Size: Variable, 1–10 elements JSON Data Type: Array of string String: 5–8 characters Values accepted: • Refer to *EMV Specification Bulletin 255*.	N/A	N/A	PRes = R	
DS Reference Number						OReq = R	
DS Transaction ID						OReq = R ORes = R	
DS URL List Field Name: `dsUrlList`	List of DS URLs to which the 3DS Server will send the AReq message. The DS optionally provides this list in case there are preferred DS URLs for some countries.	DS	Size: Variable, 1–99 elements JSON Data Type: Array of objects Values accepted: See Table A.7 for DS URL List.	N/A	N/A	PRes = O	

Data Element/ Field Name	Description	Source	Length/Format/ Values	Device Channel	Message Category	Message Inclusion	Condition Inclusion
EMV Payment Token Information Field Name: `payTokenInfo`	Information about de-tokenised Payment Token.	3DS Server DS	Length: Variable JSON Data Type: Object Refer to Table A.25 for data elements to include. Note: Data will be formatted into a JSON object prior to being placed into the EMV Payment Token field of the message.	01-APP 02-BRW 03-3RI	01-PA 02-NPA	AReq = O	
Expandable Information Text			Note: Bold text is supported in this data element and is enclosed between **. For example: “This is **bold** text” is rendered as This is **bold** text.				

Data Element/ Field Name	Description	Source	Length/Format/ Values	Device Channel	Message Category	Message Inclusion	Condition Inclusion
Information Continuation Indicator Field Name: `infoContinueIndicator`	Indicator notifying the ACS that the Cardholder selected the Information Continue button in the Information UI template. Note: The Boolean value of true is the only valid response for this field when it is present.	3DS SDK	JSON Data Type: Boolean Value accepted: • true	01-APP	01-PA 02-NPA	CReq = C	Required for ACS UI Type = 07 if the Cardholder selects the Information Continuation button on the device.
Information Continuation Label Field Name: `infoContinueLabel`	UI label used in the UI for the button that the Cardholder selects in the Information UI template.	ACS	Length: Variable, maximum 45 characters JSON Data Type: String	01-APP	01-PA 02-NPA	CRes = C	See Table A.20 for presence conditions.
Issuer Image			Values accepted: • Refer to ~~Table A.16~~Table A.21 for data elements.				
Merchant Name			Same name used in the authorisation message as defined in ISO 8583-1.				

Data Element/ Field Name	Description	Source	Length/Format/ Values	Device Channel	Message Category	Message Inclusion	Condition Inclusion
Merchant Risk Indicator		3DS Server	Refer to ~~Table A.9~~Table A.11 for data elements. Note: Data will be formatted into a JSON object prior to being placed into the ~~Device~~ Merchant Risk Indicator field of the message.				
Message Extension		3DS SDK ACS DS	~~Length~~Size: Variable, maximum ~~81920 characters~~1–15 elements JSON Data Type: Array of objects Values accepted: • Refer to Table A.~~7~~9 for data elements.			OReq = C ORes = C	
Message Type			• ORes • OReq			OReq = R ORes = R	

Data Element/ Field Name	Description	Source	Length/Format/ Values	Device Channel	Message Category	Message Inclusion	Condition Inclusion
Message Version Number			Value accepted: • Major.minor.patch Example: • 99.99.99 Refer to ~~Table 1.5~~*EMV Specification Bulletin 255.*			OReq = R ORes = R	
Multi-Transaction Field Name: `multiTransaction`	Additional transaction information in case of multiple transactions or merchants.	3DS Server	Length: Variable JSON Data Type: Object Refer to Table A.18 for data elements to include.	01-APP 02-BRW 03-3RI	01-PA 02-NPA 03-3RI	AReq = O	

Data Element/ Field Name	Description	Source	Length/Format/ Values	Device Channel	Message Category	Message Inclusion	Condition Inclusion
OOB App Label	Example: "oobAppLabel": ~~"Click here to open"~~: "Open Your Bank App"	~~N/A~~ACS				CRes = ~~O~~C	~~Note: This element has been defined to support future enhancements to the OOB message flow. An ACS will not provide this value and a 3DS SDK will not perform any processing and will not display the OOB App Label in this version of the specification.~~ Only present for ACS UI Type = 04 if: • OOB App URL Indicator = 01 in the CReq message AND • the ACS uses the OOB Authentication App automatic switching feature for this transaction.

Data Element/ Field Name	Description	Source	Length/Format/ Values	Device Channel	Message Category	Message Inclusion	Condition Inclusion
OOB App Status Field Name: `oobAppStatus`	Status code indicating the problem type encountered when using the OOB App URL.	3DS SDK	Length: 2 characters JSON Data Type: String Values accepted: • 01 = Open OOB App URL failed • 02–99 = Reserved for EMVCo future use (values invalid until defined by EMVCo)	01-APP	01-PA 02-NPA	CReq = C	Required if the Cardholder encountered an error when selecting the OOB App URL in the ACS UI Type = 04 or 06.
OOB App URL	~~Mobile Deep~~Universal App Link to an authentication app used in the ~~out-of-band~~OOB authentication. The OOB App URL will open the appropriate location within the OOB Authentication App. Refer to Table 1.3 for Universal App Link definition.	~~N/A~~ACS	Length: Variable, maximum ~~256~~ 2048 characters Value accepted: ~~Fully Qualified URL~~Universal App Link			CRes = ~~O~~C	Only present for [ACS UI Type = 04 if the OOB App Label is present OR ACS UI Type = 06] AND if: • OOB App URL Indicator = 01 in the CReq message; AND

Data Element/ Field Name	Description	Source	Length/Format/ Values	Device Channel	Message Category	Message Inclusion	Condition Inclusion
							• the ACS uses the OOB Authentication App automatic switching feature for this transaction ~~Note: this element has been defined to support future enhancements to the OOB message flow. An ACS will not provide this value and a 3DS SDK will not perform any processing of the OOB App URL in this version of the specification.~~

Data Element/ Field Name	Description	Source	Length/Format/ Values	Device Channel	Message Category	Message Inclusion	Condition Inclusion
OOB App URL Indicator Field Name: `oobAppURLInd`	Indicates if the 3DS SDK supports the OOB App URL.	3DS SDK	Length: 2 characters JSON Data Type: String Values accepted: • 01 = Supported • 02 = Not supported by the device • 03 = Not supported by the 3DS Requestor • 04–79 = Reserved for EMVCo future use (values invalid until defined by EMVCo) • 80–99 = Reserved for DS use If SDK Type = 02 and Split-SDK Type = 02, the OOB App URL Indicator is set to 02. Note: OOB App URL does not work for the Split-SDK/Browser.	01-APP	01-PA 02-NPA	CReq = R	

Data Element/ Field Name	Description	Source	Length/Format/ Values	Device Channel	Message Category	Message Inclusion	Condition Inclusion
OOB Continuation Indicator	Indicator notifying the ACS that Cardholder has selected the OOB Continuation button in an OOB authentication method, or that the 3DS SDK automatically completes without any Cardholder interaction. ~~Indicator notifying the ACS that Cardholder has completed the authentication as requested by selecting the Continue button in an Out-of-Band (OOB) authentication method.~~ ~~Note: The Boolean value of true is the only valid response for this field when it is present.~~		Length: 2 characters JSON Data Type: ~~Boolean~~String Values accepted: ~~true~~ • 01 = Cardholder clicks the button • 02 = Automatic complete • 03–99 = Reserved for EMVCo future use (values invalid until defined by EMVCo)				Required if: • ACS UI Type = 04 ~~when the Cardholder has selected that option on the device~~ unless Challenge Additional Code is present; OR • ACS UI Type = 06.

Data Element/ Field Name	Description	Source	Length/Format/ Values	Device Channel	Message Category	Message Inclusion	Condition Inclusion
OOB Continuation Label							Required ~~when~~ for ACS UI Type = 04 ~~in when the Cardholder has selected that option on the device~~ if the OOB App Label is not present. ~~Note: If present, either of the following must also be present:~~ • ~~Challenge Information Header, OR~~ • ~~Challenge Information Text~~
Operation Category Field Name: `opCategory`	Indicates the category/type of information.	DS	Length: 2 characters JSON Data Type: String Value accepted: • 01 = General • 02 = Operational alert • 03 = Public Key • 04 = LOA/AOC expiry • 05 = Fraud	N/A	N/A	OReq = R	

Data Element/ Field Name	Description	Source	Length/Format/ Values	Device Channel	Message Category	Message Inclusion	Condition Inclusion
			• 06 = Other • 07–79 = Reserved for EMVCo future use (values invalid until defined by EMVCo) • 80–99 = Reserved for DS use				
Operation Description Field Name: `opDescription`	Describes the reason for the operational communication or the response to an action taken by the recipient.	DS	Length: Variable, maximum 20000 characters JSON Data Type: String	N/A	N/A	OReq = R	
Operation Expiration Date Field Name: `opExpDate`	The date after which the relevance of the operational information (e.g., certificate expiration dates, SLAs, etc.) expires.	DS	Length: 8 characters JSON Data Type: String Format accepted: YYYYMMDD	N/A	N/A	OReq = R	

Data Element/ Field Name	Description	Source	Length/Format/ Values	Device Channel	Message Category	Message Inclusion	Condition Inclusion
Operation Message Status Field Name: `opStatus`	Indicates the status of the Operation Request message sequence from the source of the OReq.	3DS Server ACS	Length: 2 characters JSON Data Type: String Values accepted: • 01 = Successfully received messages • 02 = Message sequence is broken • 03 = Requested action is not supported or not executed by the 3DS Server or ACS when OReq message was received • 04–79 = Reserved for EMVCo future use (values invalid until defined by EMVCo) • 80–99 = Reserved for DS use	N/A	N/A	ORes = R	

Data Element/ Field Name	Description	Source	Length/Format/ Values	Device Channel	Message Category	Message Inclusion	Condition Inclusion

Operation Prior Transaction Reference
 	
Field Name: `opPriorTransRef`
	This data element provides additional information enabling the recipient to reference a prior transaction.	DS	JSON Data Type: Object
			• `transIdType`: 2 characters
			  o 01 = 3DS Server
			  o 02 = DS
			  o 03 = ACS
			• `transId`: 36 characters
			  o Canonical format as defined in IETF RFC 4122. May utilise any of the specified versions as long as the output meets specified requirements.
			
			For example, a prior DS Transaction ID would be represented as:
			`"opPriorTransRef":`
			`{`
			`"transIdType":"02",`
			`"transId":"4317fdc3-ad24-5443-8000-000000000891"`
			`}`
				N/A	N/A	OReq = O	

Data Element/ Field Name	Description	Source	Length/Format/ Values	Device Channel	Message Category	Message Inclusion	Condition Inclusion

Operation Sequence
Field Name: `opSeq`
	Indicates the current and total messages in an OReq message sequence.
	`seqId`: This element uniquely identifies a message sequence and will remain constant in the sequence of messages.
	`seqNum`: This element represents the current message in the sequence.
	`seqTotal`: This element represents the total number of messages in the sequence and will remain constant in the sequence of messages.
		DS	JSON Data Type: Object
			• `seqId`: 36 characters
			o Canonical format as defined in IETF RFC 4122. May utilise any of the specified versions as long as the output meets specified requirements.
			• `seqNum`: 2 characters
			Values accepted:
			o 01–99
			• `seqTotal`: 2 characters
			Values accepted:
			o 01–99
			For example, the first out of three messages in an OReq sequence would be represented as:
			`"opSeq":{`
			`"seqId":"4317fdc3-ad24-5443-8000-000000000891",`
				N/A	N/A	OReq = R	

Data Element/ Field Name	Description	Source	Length/Format/ Values	Device Channel	Message Category	Message Inclusion	Condition Inclusion
			"seqNum":"01", "seqTotal":"03"}				
Operation Severity Field Name: opSeverity	Indicates the importance/ severity level of the operational information. Critical = Immediate action to be taken by recipient Major = Major impact; Upcoming action to be taken by recipient Minor = Minor impact; Upcoming action to be taken by recipient Informational = Informational only with no immediate action by recipient	DS	Length: 2 characters JSON Data Type: String Value accepted: • 01 = Critical • 02 = Major • 03 = Minor • 04 = Informational • 05–79 = Reserved for EMVCo future use (values invalid until defined by EMVCo) • 80–99 = Reserved for DS use	N/A	N/A	OReq = R	
Payee Origin Field Name: payeeOrigin	The origin of the payee that will be provided in the SPC Transaction Data Refer to Secure Payment Confirmation.	3DS Server	Length: Variable, maximum 2048 characters JSON Data Type: String Value accepted: • Fully Qualified URL	02-BRW	01-PA 02-NPA	AReq = C	Required if 3DS Requestor SPC Support = Y
Payment System Image			Refer to Table ~~A.17~~A.22 for data elements.				

Data Element/ Field Name	Description	Source	Length/Format/ Values	Device Channel	Message Category	Message Inclusion	Condition Inclusion
Purchase Amount							• Required for 02-NPA if 3DS Requestor Authentication Indicator = 02, ~~or~~ 03, 07, 08, 09 • Required for 02-NPA if 3RI Indicator = 01, 02, 06, 07, 08, 09, 11, 15
Purchase Currency							• Required for 02-NPA if 3DS Requestor Authentication Indicator = 02, ~~or~~ 03, 07, 08, 09 • Required for 02-NPA if 3RI Indicator = 01, 02, 06, 07, 08, 09, 11, 15

Data Element/ Field Name	Description	Source	Length/Format/ Values	Device Channel	Message Category	Message Inclusion	Condition Inclusion
Purchase Currency Exponent	• ~~Yen~~JPY = 0						• Required for 02-NPA if 3DS Requestor Authentication Indicator = 02, ~~or~~ 03, 07, 08, 09 • Required for 02-NPA if 3RI Indicator = 01, 02, 06, 07, 08, 09, 11, 15
Purchase Date & Time	Date and time of the ~~purchase~~authentication ~~expressed~~ converted into UTC.						• Required for 02-NPA if 3DS Requestor Authentication Indicator = 02, ~~or~~ 03, 07, 08, 09 • Required for 02-NPA if 3RI Indicator = 01, 02, 06, 07, 08, 09, 11, 15
Read Order Field Name: `readOrder`	Indicates the order in which to process the card range records from the PRes message.	DS	Length: 2 characters JSON Data Type: String Values accepted:	N/A	N/A	PRes = R	

Data Element/ Field Name	Description	Source	Length/Format/ Values	Device Channel	Message Category	Message Inclusion	Condition Inclusion
			• 01 = Direct order/FIFO (First In First Out) • 02 = Reverse order/LIFO (Last In First Out • 03–79 = Reserved for EMVCo future use (values invalid until defined by EMVCo) • 80–99 = Reserved for DS use				
Recurring Amount Field Name: `recurringAmount`	Recurring amount in minor units of currency with all punctuation removed.	3DS Server	Length: Variable, maximum 48 characters JSON Data Type: String Example: Purchase amount is USD 123.45 Example values accepted: • 12345 • 012345 • 0012345	01-APP 02-BRW 03-3RI	01-PA 02-NPA	AReq = C	• Required if: [3DS Requestor Authentication Indicator = 02 or 03; OR • 3RI Indicator = 01 or 02] AND • Recurring Indicator/ Amount Indicator = 01

Data Element/ Field Name	Description	Source	Length/Format/ Values	Device Channel	Message Category	Message Inclusion	Condition Inclusion
Recurring Currency Field Name: `recurringCurrency`	Currency in which the Recurring Amount is expressed.	3DS Server	Length: 3 characters; Numeric JSON Data Type: String Values accepted: • ISO 4217 three-digit currency code, other than those listed in Table A.5.	01-APP 02-BRW 03-3RI	01-PA 02-NPA	AReq = C	Required if the Recurring Amount is present.
Recurring Currency Exponent Field Name: `recurringExponent`	Minor units of currency as specified in the ISO 4217 currency exponent. Example: • USD = 2 • JPY = 0	3DS Server	Length: 1 character; Numeric JSON Data Type: String	01-APP 02-BRW 03-3RI	01-PA 02-NPA	AReq = C	Required if the Recurring Amount is present.
Recurring Date Field Name: `recurringDate`	Effective date of new authorised amount following first/promotional payment in recurring or instalment transaction.	3DS Server	Length: 8 characters JSON Data Type: String Date format accepted: YYYYMMDD	01-APP 02-BRW 03-3RI	01-PA 02-NPA	AReq = C	Required if Recurring Indicator/ Frequency Indicator = 01.
Recurring Expiry							• ~~Required if 3DS Requestor Authentication Indicator = 02 or 03.~~

Data Element/ Field Name	Description	Source	Length/Format/ Values	Device Channel	Message Category	Message Inclusion	Condition Inclusion
							~~• Required for 03-3RI if 3RI Indicator = 01 or 02.~~ Required if there is an end date.
Recurring Frequency	Indicates the minimum number of days between authorisations for a recurring or instalment transaction.		Values accepted: • Numeric values between 1 and 9999				~~• Required if 3DS Requestor Authentication Indicator = 02 or 03.~~ ~~• Required for 03-3RI if 3RI Indicator = 01 or 02.~~ Required if Recurring Indicator/ Frequency Indicator = 01.
Recurring Indicator Field Name: `recurringInd`	Indicates whether the recurring or instalment payment has a fixed or variable amount and frequency. The Recurring Indicator object contains: • the Amount Indicator	3DS Server	JSON Data Type: Object **Amount Indicator** Field Name: `amountInd` Values accepted: • 01 = Fixed Purchase Amount	01-APP 02-BRW 03-3RI	01-PA 02-NPA	AReq = C	Required if: • 3DS Requestor Authentication Indicator = 02 or 03; OR • 3RI Indicator = 01 or 02.

Data Element/ Field Name	Description	Source	Length/Format/ Values	Device Channel	Message Category	Message Inclusion	Condition Inclusion
	• the Frequency Indicator Example: `{"recurringInd":{"amountInd":"01","frequencyInd":"02"}}`		• 02 = Variable Purchase Amount • 03–79 = Reserved for EMVCo future use (values invalid until defined by EMVCo) • 80–99 = Reserved for DS use **Frequency Indicator** Field Name: `frequencyInd` Values accepted: • 01 = Fixed Frequency • 02 = Variable or Unknown Frequency • 03–79 = Reserved for EMVCo future use (values invalid until defined by EMVCo) • 80–99 = Reserved for DS use				

Data Element/ Field Name	Description	Source	Length/Format/ Values	Device Channel	Message Category	Message Inclusion	Condition Inclusion
Resend Challenge Information Code	Indicator to the ACS that the Cardholder selected the Resend Information button~~to resend the challenge information code to the Cardholder~~.		Value accepted: ~~N = Do not Resend~~				Required ~~for Native UI~~ if the Cardholder is requesting the ACS to resend challenge information (value = Y) AND ACS UI Type = 01.
Resend Information Label							See Table A.20 for inclusion conditions.~~Required for Native UI if the ACS is allowing the Cardholder to request resending authentication information.~~
Results Message Status			• 04 = 3DS Server will process Decoupled Authentication in a subsequent authentication • 05–79 = Reserved for EMVCo future use (values invalid until defined by EMVCo)				

Data Element/ Field Name	Description	Source	Length/Format/ Values	Device Channel	Message Category	Message Inclusion	Condition Inclusion
SDK App ID	Note: In case of Split-SDK/Browser, the SDK App ID value is not reliable, and may change for each transaction.						
SDK Counter SDK to ACS	Note: The counter is the decimal value equivalent of the byte, encoded as a numeric string.		Values accepted: • 000–255				
SDK Encrypted Data	~~Note: This element is the only field encrypted in this version of the EMV 3-D Secure specification.~~						
SDK Reference Number	Identifies the vendor and version ~~for~~ of the 3DS SDK that is ~~integrated in a 3DS Requestor App,~~utilised for a specific transaction. The value is assigned by EMVCo when the Letter of Approval of the specific 3DS SDK is ~~approved~~ issued.						

Data Element/ Field Name	Description	Source	Length/Format/ Values	Device Channel	Message Category	Message Inclusion	Condition Inclusion
SDK Server Signed Content Field Name: sdkServerSignedContent	Contains the JWS object (represented as a string) created by the Split-SDK Server for the AReq message. See Section 6.2.2.3 for details.	3DS SDK	Length: Variable JSON Data Type: String Value accepted: The body of JWS object (represented as a string) will contain the following data elements as defined in Table A.1: • SDK Reference Number • SDK Signature Timestamp • SDK Transaction ID • Split-SDK Server ID	01-APP	01-PA 02-NPA	AReq = C	Required if SDK Type = 02.
SDK Signature Timestamp Field Name: sdkSignatureTimestamp	Date and time indicating when the 3DS SDK generated the Split-SDK Server Signed Content converted into UTC.	3DS SDK	Length: 14 characters JSON Data Type: String Date format accepted: • YYYYMMDDHHMM	01-APP	01-PA 02-NPA	See SDK Server Signed Content.	See SDK Server Signed Content.

Data Element/ Field Name	Description	Source	Length/Format/ Values	Device Channel	Message Category	Message Inclusion	Condition Inclusion
SDK Type Field Name: `sdkType`	Indicates the type of 3DS SDK. This data element provides additional information to the DS and ACS to determine the best approach for handling the transaction.	3DS Server	Length: 2 characters JSON Data Type: String Values accepted: • 01 = Default SDK • 02 = Split-SDK • 03–79 = Reserved for EMVCo future use (values invalid until defined by EMVCo) • 80–99 = Reserved for DS use	01-APP	01-PA 02-NPA	AReq = R	
Seller Information Field Name: `sellerInfo`	Additional transaction information for transactions where merchants submit transaction details on behalf of another entity, i.e. individual sellers in a marketplace or drivers in a ride share platform.	3DS Server	Length: Variable, 1–50 elements JSON Data Type: Array of objects Refer to Table A.19 for data elements to include.	01-APP 02-BRW 03-3RI	01-PA 02-NPA	AReq = O	
Serial Number	*The following Note was added at the end of the description:*					PRes = ~~O~~C	PRes: Absent if the Card Range Data File URL is present.

Data Element/ Field Name	Description	Source	Length/Format/ Values	Device Channel	Message Category	Message Inclusion	Condition Inclusion
	Note: Serial Number is not provided when the DS and the 3DS Server select the Card Range Data File download option.						
SPC Incompletion Indicator Field Name: `spcIncompInd`	Reason that the SPC authentication was not completed.	3DS Server	Length: 2 characters JSON Data Type: String Values accepted: • 01 = SPC did not run or did not successfully complete • 02 = Cardholder cancels the SPC authentication • 03 = SPC timed out • 04–99 = Reserved for EMVCo future use (values invalid until defined by EMVCo)	02-BRW	01-PA 02-NPA	AReq = C	Required if the 3DS Requestor attempts to invoke SPC API and there is an error.

Data Element/ Field Name	Description	Source	Length/Format/ Values	Device Channel	Message Category	Message Inclusion	Condition Inclusion
SPC Transaction Data Field Name: `spcTransData`	Information that the 3DS Requestor passes in the SPC API for display in the Smart Modal Window	ACS DS	JSON Data Type: Object Values accepted: • Refer to Table A.28 for data elements. Note: For NPA, Amount is set to 0 and Currency is set to any valid value.	02-BRW	01-PA 02-NPA	ARes = C	Required if Transaction Status = S.
Split-SDK Server ID Field Name: `splitSdkServerID`	DS assigned Split-SDK Server identifier. Each DS can provide a unique ID to each Split-SDK Server on an individual basis.	Split-SDK Server	Length: Variable, maximum 32 characters JSON Data Type: String Value accepted: Any individual DS may impose specific formatting and character requirements on the contents of this field.	01-APP	01-PA 02-NPA	See SDK Server Signed Content	See SDK Server Signed Content.

Data Element/ Field Name	Description	Source	Length/Format/ Values	Device Channel	Message Category	Message Inclusion	Condition Inclusion

Split-SDK Type
Field Name: `splitSdkType`
	Indicates the characteristics of a Split-SDK.
	Split-SDK Variant: Implementation characteristics of the Split-SDK client
	Limited Split-SDK Indicator: If the Split-SDK client has limited capabilities
	Example:
	`"splitSdkType":{`
	`"sdkVariant":"01",`
	`"limitedInd":"Y"`
	`}`
		3DS Server	Length: Variable
			JSON Data Type: Object
			`sdkVariant`
			Length: 2 characters
			JSON Data Type: String
			Values accepted:
			• 01 = Native Client
			• 02 = Browser
			• 03 = Shell
			• 04–79 = Reserved for EMVCo future use (values invalid until defined by EMVCo)
			• 80–99 = Reserved for DS use
			`limitedInd`
			Length: 1 character
			JSON Data Type: String
			Value accepted:
			• Y = Limited
			Only present if value = Y
				01-APP	01-PA
					02-NPA
						AReq = C	Required if SDK Type = 02

Data Element/ Field Name	Description	Source	Length/Format/ Values	Device Channel	Message Category	Message Inclusion	Condition Inclusion
Submit Authentication Label							~~Refer to Table A.18 for additional information.~~Required if ACS UI Type = 01, 02 or 03.
Tax ID Field Name: `taxId`	Cardholder's tax identification.	3DS Server	Length: Variable maximum 45 characters JSON Data Type: String	01-APP 02-BRW 03-3RI	01-PA 02-NPA	AReq = C	Conditional based on DS rules.
Toggle Position Indicator Field Name: `togglePositionInd`	Indicates if the Trust List and/or Device Binding prompt should be presented below or above the action buttons (Submit Authentication, OOB App, OOB Continuation, Information Continuation, Challenge Additional).	ACS	Length: 2 characters JSON Data Type: String Values accepted: • 01 = Above the buttons Only present if value = 01 If the Toggle Position Indicator is not present, the Trust List or Device Binding are below the action buttons. • 02–99 = Reserved for EMVCo future use (values invalid until defined by EMVCo)	01-APP	01-PA 02-NPA	CRes = O	

Data Element/ Field Name	Description	Source	Length/Format/ Values	Device Channel	Message Category	Message Inclusion	Condition Inclusion
Transaction Challenge Exemption Field Name: `transChallengeExemption`	Exemption applied by the ACS to authenticate the transaction without requesting a challenge.	ACS	Length: 2 characters JSON Data Type: String Values accepted: • 05 = Transaction Risk Analysis exemption • 08 = Trust List exemption • 10 = Low Value exemption • 11 = Secure Corporate Payments exemption • 79 = No exemption applied • 01–04, 06, 07, 09 and 12–78 = Reserved for EMVCo future use (values invalid until defined by EMVCo) • 80–99 = Reserved for DS use	01-PA 02-NPA 03-3RI	01-APP 02-BRW	ARes = O	

Data Element/ Field Name	Description	Source	Length/Format/ Values	Device Channel	Message Category	Message Inclusion	Condition Inclusion
Transaction Status	~~Note:~~ The Final CRes message can only contain ~~only~~ a value of Y or N or D. ~~Note: If the 3DS Requestor Challenge Indicator = 06 (No challenge requested; Data share only), then a Transaction Status of C is not valid.~~ Transaction Status = C or S is not allowed for Device Channel = 3RI.		• S = Challenge using SPC				For 01-PA see Table A.~~15~~17 for Transaction Status presence conditions. For 02-NPA, ~~Conditional as defined by the DS.~~requirements for the presence and values of Transaction Status are DS-specific.
Transaction Status Reason			• 27 = Preferred Authentication Method not supported • 28 = Validation of content security policy failed • 29 = Authentication attempted but not completed by the Cardholder. Fall back to Decoupled Authentication				For 02-NPA, ~~Conditional as defined by the DS.~~requirements for the presence and values of Transaction Status are DS-specific.

Data Element/ Field Name	Description	Source	Length/Format/ Values	Device Channel	Message Category	Message Inclusion	Condition Inclusion
			• 30 = Authentication completed successfully but additional authentication of the Cardholder required. Reinitiate as Decoupled Authentication • 31–79 = Reserved for EMVCo future use (values invalid until defined by EMVCo)				
Transaction Status Reason Information Field Name: transStatusReasonInfo	Provides additional information on the Transaction Status Reason.	ACS DS	Length: Variable, maximum 256 characters JSON Data Type: String	01-APP 02-BRW 03-3RI	01-PA 02-NPA	ARes = O RReq = O	
Transaction Type			Note: Values derived from ~~the~~ ISO 8583-1 ~~Standard~~.				
~~Whitelisting~~Trust List Data Entry Field Name: ~~whitelisting~~trustListDataEntry	Indicator provided by the 3DS SDK to the ACS to confirm whether ~~whitelisting was opted by the cardholder~~ the Cardholder gives consent to Trust List.	3DS SDK	Length: 1 character JSON Data Type: String Values accepted: • Y = ~~Whitelisting Confirmed~~Consent given to Whitelist	01-APP	01-PA 02-NPA	CReq = C	Required if: • Trust List Information Text was present in the preceding CRes message AND

Data Element/ Field Name	Description	Source	Length/Format/ Values	Device Channel	Message Category	Message Inclusion	Condition Inclusion
			• N = ~~Whitelisting Not Confirmed~~ Consent not given to Trust List Note: If the Cardholder action changes the default value, then the value = Y. Otherwise the value = N.				• Challenge Cancelation Indicator is not present. ~~If Whitelisting Information Text was present in the CRes message, SDK must provide this data element to the ACS in the CReq message.~~
~~Whitelisting~~Trust List Information Text Field Name: ~~whitelisting~~trustListInfoText	Text provided by the ACS~~/Issuer~~ to the Cardholder during a ~~Whitelisting~~Trust List transaction. Example: • "Would you like to add this Merchant to your ~~whitelist~~Trust List?"	ACS	Length: Variable, maximum 64 characters	01-APP	01-PA 02- NPA	CRes = O	~~If present, must be displayed by the SDK.~~See Table A.20 for presence conditions.

Data Element/ Field Name	Description	Source	Length/Format/ Values	Device Channel	Message Category	Message Inclusion	Condition Inclusion
~~Whitelist~~Trust List Status Field Name: ~~white~~trustListStatus	Enables the communication of ~~trusted beneficiary/whitelist~~trust list status between the ACS, the DS and the 3DS Requestor.		• Y = 3DS Requestor is ~~whitelisted~~ Trust Listed by Cardholder • N = 3DS Requestor is not ~~whitelisted~~ Trust Listed by Cardholder • U = ~~Whitelist~~Trust List status unknown, unavailable, or does not apply				
~~Whitelist~~Trust List Status Source Field Name: ~~white~~trustListStatusSource	This data element will be populated by the system setting ~~Whitelist~~ Trust List Status.						Required if ~~Whitelist~~ Trust List is present.
WebAuthn Credential List Field Name: webAuthnCredList	List of credential IDs registered for the Cardholder Account Number.	ACS	Size: Variable, 1–10 elements JSON Data Type: Array of objects The object contains: • Relying Party ID Field Name: rpID Length: Variable, maximum 2048 characters	02-BRW	01-PA 02-NPA	ARes = C	Required if Transaction Status = S

Data Element/ Field Name	Description	Source	Length/Format/ Values	Device Channel	Message Category	Message Inclusion	Condition Inclusion
			• WebAuthn Credential Field Name: `credentialIds` Length: Variable, 16–1000 characters Base64url-encoded				
Why Information Text			Note: Bold text is supported in this data element and is enclosed between **. For example: “This is **bold** text” is rendered as This is **bold** text.				

~~A.5 Detailed Field Values~~

The following sections provide additional details on the values for some data elements listed in Table A.1

The headings in Section A.5.1 and A.5.2 were renumbered, with no change to their content.

~~A.5.1~~A.5 Device Information—01-APP Only

~~A.5.2~~A.6 Browser Information—02-BRW Only

~~A.5.3~~A.7 3DS Method Data

Clarification was added in the introduction to correct a typographical error.

The data is exchanged between the 3DS Requestor and the ACS via the Cardholder Browser.

Table A.2: 3DS Method Data

Data Element/Field Name	Description	Length/Format/Values	Recipient	Message Category	Message Inclusion
3DS Method Notification URL		Length: Variable, maximum 2048 characters JSON Data Type: String Value accepted: • Fully Qualified URL			
3DS Server Transaction ID		Refer to 3DS Server Transaction ID in Table A.1.			

~~A.5.4~~A.8 Browser CReq and CRes POST

Table A.3: 3DS CReq/CRes POST Data

Data Element / Field Name	Description	Recipient	Length/Format/Values	Message Inclusion
3DS Requestor Session Data	The 3DS Requestor may provide the 3DS Requestor Session Data with the CReq message to the ACS. The 3DS Requestor Session Data is optionally used to accommodate the different methods that 3DS Requestor systems use to handle session information. 3~~DS Requestor session data that is returned by the ACS in the CRes message POST to the 3DS Requestor. Optionally used to accommodate the different methods 3DS Requestor systems handle session information.~~ The ACS returns the 3DS Requestor session data with the CRes message POST to the 3DS Requestor. If the 3DS Requestor system can associate the final post with the original session without further assistance, the 3DS Requestor Session Data field may be missing. If provided by the 3DS Requestor, the Session Data must be returned by the ACS.		Length: Variable, maximum 1024 characters	• O in HTML form with the CReq message • R in HTML form with the CRes message if received with the CReq message
CReq			Base64url-encoded	

Browser CReq–CRes Data Examples

- **Example 1:** `threeDSSessionData` sent by the 3DS Requestor in the CReq message to the ACS

```
3DS Requestor Session Data from the 3DS Requestor = "merchant.com-ID-adac2434-df78-4bfa-bcd9-11ca4ccd5dca"

3DS Requestor Session Data base64URL encoded =
"bWVyY2hhbnQuY29tLUlELWFkYWMyNDM0LWRmNzgtNGJmYS1iY2Q5LTExY2E0Y2NkNWRjYQ"
```

CReq message


```
{
     "threeDSServerTransID":"8a880dc0-d2d2-4067-bcb1-b08d1690b26e",
     "acsTransID":"d7c1ee99-9478-44a6-b1f2-391e29c6b340",
     "threeDSRequestorURL":"https://merchant.com/url",
     "messageType":"CReq",
     "messageVersion":"2.3.1"
}
```


CReq message base64URL encoded


```
"eyJ0aHJlZURTU2VydmVyVHJhbnNJRCI6IjhhODgwZGMwLWQyZDItNDA2Ny1iY2IxLWIwOGQxNjkwYjI2ZSIsCSJhY3NUcmFuc0lEIjoiZDd
jMWVlOTktOTQ3OC00NGE2LWIxZjItMzkxZTI5YzZiMzQwIiwidGhyZWVEU1JlcXVlc3RvclVybCI6Imh0dHBzOi8vbWVyY2hhbnQuY29tL3V
ybCIsIm1lc3NhZ2VUeXBlIjoiQ1JlcSIsIm1lc3NhZ2VWZXJzaW9uIjoiMi4zLjAifQ"
```


HTML form


```
"htmlCreq": "<form action=\'https://acs.com.creq\' method=\'post\'>
<input type=\'hidden\' name=\'creq\' value=\
'eyJ0aHJlZURTU2VydmVyVHJhbnNJRCI6IjhhODgwZGMwLWQyZDItNDA2Ny1iY2IxLWIwOGQxNjkwYjI2ZSIsCSJhY3NUcmFuc0lEIjoiZDd
jMWVlOTktOTQ3OC00NGE2LWIxZjItMzkxZTI5YzZiMzQwIiwidGhyZWVEU1JlcXVlc3RvclVybCI6Imh0dHBzOi8vbWVyY2hhbnQuY29tL3V
ybCIsIm1lc3NhZ2VUeXBlIjoiQ1JlcSIsIm1lc3NhZ2VWZXJzaW9uIjoiMi4zLjAifQ' />
<input type=\'hidden\' name=\'threeDSSessionData\' value=\'b\'
bWVyY2hhbnQuY29tLUlELWFkYWMyNDM0LWRmNzgtNGJmYS1iY2Q5LTExY2E0Y2NkNWRjYQ'\' /></form>"
```

- **Example 2:** `threeDSSessionData` sent by the ACS in the CRes message to the 3DS Requestor

```
Base64url decoded 3DS Requestor Session Data: "merchant.com-ID-adac2434-df78-4bfa-bcd9-11ca4ccd5dca"

CRes message

{
     "threeDSServerTransID":"8a880dc0-d2d2-4067-bcb1-b08d1690b26e",
     "acsTransID":"d7c1ee99-9478-44a6-b1f2-391e29c6b340",
     "transStatus":"Y",
     "messageType":"CRes",
     "messageVersion":"2.3.1
}

Base64 URL encoded CRes message

"eyJ0aHJlZURTU2VydmVyVHJhbnNJRCI6IjhhODgwZGMwLWQyZDItNDA2Ny1iY2IxLWIwOGQxNjkwYjI2ZSIsImFjc1RyYW5zSUQiOiJkN2M
xZWU5OS05NDc4LTQ0YTYtYjFmMi0zOTFlMjljNmIzNDAiLCJ0cmFuc1N0YXR1cyI6IlkiLCJtZXNzYWdlVHlwZSI6IkNSZXMiLCJtZXNzYWd
lVmVyc2lvbiI6IjIuMy4wIix9"

3DS Requestor Session Data base64URL encoded =
"bWVyY2hhbnQuY29tLUlELWFkYWMyNDM0LWRmNzgtNGJmYS1iY2Q5LTExY2E0Y2NkNWRjYQ"

HTML form

"htmlCres": "<form action=\'https://3dss.com.cres\' method=\'post\'>
```

```
<input type=\'hidden\' name=\'cres\'
value=\'eyJ0aHJlZURTU2VydmVyVHJhbnNJRCI6IjhhODgwZGMwLWQyZDItNDA2Ny1iY2IxLWIwOGQxNjkwYjI2ZSIsImFjc1RyYW5zSUQi
OiJkN2MxZWU5OS05NDc4LTQ0YTYtYjFmMi0zOTFlMjljNmIzNDAiLCJ0cmFuc1N0YXR1cyI6IlkiLCJtZXNzYWdlVHlwZSI6IkNSZXMiLCJt
ZXNzYWdlVmVyc2lvbiI6IjIuMy4wIix9'/>
<input type=\'hidden\' name=\'threeDSSessionData\' value=\'b\'
bWVyY2hhbnQuY29tLUlELWFkYWMyNDM0LWRmNzgtNGJmYS1iY2Q5LTExY2E0Y2NkNWRjYQ'\' /></form>"
```

~~A.5.5~~A.9 Error Code, Error Description, and Error Detail

Table A.4 Error Code, Error Description, and Error Detail

Value	Error Code	Error Description	Error Detail
101	Message Received Invalid	One of the following: Message is not AReq, ARes, CReq, CRes, PReq, PRes, OReq, ORes, RReq, or RRes	
102	Message Version Number Not Supported	One of the following: • Message Version Number received is not valid for the receiving component. • Error in the Message Version Number in the Card Range Data. • Message Version Number provided for a Payment Token is not supported by the actual PAN when de-tokenised.	• Message Version Number in the Card Range Data is not active. • Message Version Number received is not supported by the receiving component. • Note: All supported Protocol Version Numbers are provided in a comma-delimited list. ~~All supported Protocol Version Numbers in a comma-delimited list.~~
203	Format or value of one or more Data Elements is Invalid according to the Specification	• UTC date and time data element is not using UTC.	

Value	Error Code	Error Description	Error Detail
205	Card Range Overlap	Overlap in the card ranges provided by the DS in the PRes message. For example, the two card ranges 11000–15000 and 13000–17000 overlap from 13000–15000.	List of Card Ranges that overlap.
206	Card Range Action Indicator	Action is not possible for the card range. For example, Delete or Modify a card range that does not exist, or Add an already existing card range.	List the Card Range and Action Indicator that is causing the error.
207	Value in the Reserved Value range	Data Element value is in the range of "Reserved for DS use" or "Reserved for EMVCo future use" and is not recognised.	Name of invalid element(s); if more than one invalid data element is detected, this is a comma-delimited list.
301	Transaction ID Not Recognised		The Transaction ID received was invalid. Invalid meaning Transaction ID not recognised~~, or Transaction ID is recognised as a duplicate~~.
305	Transaction Data Not Valid	If in response to a CReq, and a CReq message was incorrectly sent: ~~CReq message with this ACS Transaction ID has already been received and processed~~	
308	Signature Verification Failure	SDK Server Signed Content could not be verified.	Description of the failure.
309	Validation Against Content Security Policies Failure	Validation against content security policies failed.	For example, which element prevented successful validation.

Value	Error Code	Error Description	Error Detail
310	Incorrect Cryptographic Algorithm	The use of a specific cryptographic algorithm is not allowed in the specific context.	For example, which cryptographic algorithm was expected.
311	Incorrect kid	The DS detects an error for the key identifier (kid) present in the SDK Encrypted Data.	For example: • The provided kid is not recognised • The kid is not present
312	Duplicate message	A message with the same Transaction ID was already received.	The Transaction ID is recognised as a duplicate. For example, the DS receives multiple RReq messages with the same Transaction ID.
313	Inconsistent RReq message	An RReq message is received although there was no challenge (Transaction Status not equal to C or D or S) for this transaction.	The ACS sends an RReq message but the Transaction Status in the corresponding ARes message was not = C or D or S.
314	Multiple CReq messages not supported	During a challenge for the Browser flow, the ACS does not accept multiple CReq messages.	The Cardholder requests a Browser page refresh during a challenge, the 3DS Server sends a second CReq message to the ACS.
315	CReq message received after the RReq message	During a challenge for the Browser flow, the ACS receives a CReq message, after having sent the RReq message.	The Cardholder requests a Browser page refresh during a challenge after the ACS has sent the RReq message to complete the transaction.

~~A.5.6~~A.10 Excluded ISO Currency and Country Code Values

ISO Code	Numeric ~~V~~value not permitted for 3-D Secure	Alphabetic value not permitted for 3-D Secure	Definition
ISO 4217	955	XBA	European Composite Unit
ISO 4217	956	XBB	European Monetary Unit
ISO 4217	957	XBC	European Unit of Account 9
ISO 4217	958	XBD	European Unit of Account 17
ISO 4217	959	XAU	Gold
ISO 4217	960	XDR	I.M.F.
ISO 4217	961	XAG	Silver
ISO 4217	962	XPT	Platinum
ISO 4217	963	XTS	Reserved for testing
ISO 4217	964	XPD	Palladium
ISO 4217	999	XXX	No currency is involved
ISO 3166-1	901–999		Reserved by ISO to designate country names not otherwise defined

~~A.5.7~~A.11 Card Range Data

The Card Range Data data element contains information returned in ~~the~~a PRes message to the 3DS Server from the specific DS that indicates the most recent EMV 3-D Secure versions supported by the ACS that hosts that card range. ~~It also~~Card Range Data may optionally also contain the ACS URL for the 3DS Method if supported by the ACS Protocol Version and the DS ~~Start and End~~ Protocol ~~Versions~~ Version list which support th~~e~~at card range. The detailed data elements are outlined in Table ~~A.1~~A.6.

Note: ~~There may be as many~~The Card Range Data is an array containing as many JSON objects as there are stored card ranges in the DS being called.

Table A.6 Card Range Data

Data Element/Field Name	Description	Length/Format/Values	Inclusion
Ranges Field Name: `ranges`	The Ranges array contains the Start Range and End Range. It contains one or more card ranges. Refer to the following elements: • Start Range • End Range	Size: Variable, maximum 1–5000 elements JSON Data Type: Array of objects	R
Start ~~Card~~ Range Field Name: `start~~Range~~`	Start of the card range.	Length: 13–19 characters JSON Data Type: String	R
End ~~Card~~ Range Field Name: `end~~Range~~`	End of the card range.	Length: 13–19 characters JSON Data Type: String	R

Data Element/Field Name	Description	Length/Format/Values	Inclusion
Action Indicator	~~The card ranges are processed in the order returned.~~ ~~Note: If the Serial Number is not included in the PReq message, then the action is A = Add for all card ranges returned (the Action Indicator is ignored in the PRes message).~~		
Issuer Country Code Field Name: `issuerCountryCode`	Qualifies the Issuer country for the Ranges.	Length: 3 characters JSON Data Type: String Value accepted: • ISO 3166-1 numeric three-digit country code, other than exceptions listed in Table A.5.	O
DS Protocol Versions Field Name: `dsProtocolVersion`	Contains the list of active protocol versions supported by the DS. If the DS Protocol Version is present in the Card Range data element, it overrides the DS Protocol Versions in the PRes message.	Size: Variable, 1–10 elements JSON Data Type: Array of string String: 5–8 characters Values accepted: • Refer to *EMV Specification Bulletin 255*	O
~~ACS End Protocol Version~~ ~~Field Name:~~ ~~`acsEndProtocolVersion`~~	~~The most recent active protocol version that is supported for the ACS URL. Refer to Table 1.5 for active protocol version numbers.~~	~~Length: Variable, 5–8 characters~~ ~~JSON Data Type: String~~ ~~Note: If the ACS End Protocol Version is not available, this value is the DS End Protocol Version for that card range.~~	~~R~~

Data Element/Field Name	Description	Length/Format/Values	Inclusion

ACS Protocol Versions
Field Name: `acsProtocolVersions`
	Array of objects containing the list of protocol versions supported by the ACS for the card range, with their associated ACS Information Indicator, the 3DS Method URL and the list of Supported Message Extensions.
	• Version
	• ACS Information Indicator
	• 3DS Method URL
	• Supported Message Extension
		Size: Variable, maximum 1–10 elements
		JSON Data Type: Array of objects
		Values accepted:
		• Refer to the data elements:
		  ○ Version
		  ○ ACS Information Indicator
		  ○ 3DS Method URL
		  ○ Supported Message Extension
			R

Version Field Name: `version`	The Protocol Version supported by the ACS for the card range.	Length: 5–8 characters JSON Data Type: String Values accepted: • Refer to *EMV Specification Bulletin 255*	R

Data Element/Field Name	Description	Length/Format/Values	Inclusion

ACS Information Indicator
 	
Field Name:
acsInfoInd
	Provides additional information for a particular protocol version to the 3DS Server. The element lists all applicable values for the card range.
	
	Example:
	
	{
	"acsInfoInd":["01","02","03","04","05","06","07"]
	}
		~~Length: 2 characters~~ Size: Variable, maximum 1–99 elements
		
		String: 2 characters
		
		~~String v~~Values accepted:
		• 04 = ~~Whitelisting~~Trust List Supported
		• 05 = Device Binding Supported
		• 06 = WebAuthn Authentication Supported
		• 07 = SPC Authentication Supported
		• 08 = Transaction Risk Analysis Exemption Supported
		• 09 = Trust List Exemption Supported
		• 10 = Low Value Exemption Supported
		• 11 = Secure Corporate Payments Exemption Supported
		• ~~05~~12–79 = Reserved for EMVCo future use (values invalid until defined by EMVCo)	

~~ACS Start Protocol Version~~ ~~Field Name:~~ ~~acsStartProtocolVersion~~	~~The earliest (i.e. oldest) active protocol version that is supported by the ACS.~~ ~~Refer to Table 1.5 for active protocol version numbers.~~	~~Length: Variable, 5–8 characters~~ ~~JSON Data Type: String~~ ~~Note: If the ACS Start Protocol Version is not available, this value is the DS Start Protocol Version for that card range.~~	~~R~~
~~DS End Protocol Versions~~ ~~Field Name:~~ ~~dsEndProtocolVersion~~	~~The most recent active protocol version this is supported by the DS.~~	~~Length: Variable, 5–8 characters~~ ~~JSON Data Type: String~~	~~O~~
~~DS Start Protocol Version~~	~~The earliest (i.e. oldest) active protocol version that is supported by the DS.~~	~~Length: Variable, 5–8 characters~~	~~O~~

Data Element/Field Name	Description	Length/Format/Values	Inclusion
~~Field Name: dsStartProtocolVersion~~		~~JSON Data Type: String~~	
3DS Method URL	The ACS URL that will be used by the 3DS Method for a particular protocol version.	Length: Variable, maximum ~~256~~ 2048 characters	
Supported Message Extension Field Name: supportedMsgExt	List of message extensions supported by the ACS that contains the Assigned Extension Group Identifier and the Extension Version Number.	Size: Variable, maximum 1–15 elements JSON Data Type: Array of objects Value accepted: • Refer to Table A.8 o Assigned Extension Group Identifier o Extension Version Number	C Present if not empty

Previously placed at the end of Section A.11, the following Card Range Data Example was moved, and it now directly follows Table A.6.

Card Range Data Example

```
{"cardRangeData": [
                    {"ranges": [
      {"start": "1000000000000000",
       "end": "1000000000005000"},
      {"start": "1000000000006000",
       "end": "1000000000007000"}
                    ],
                    "actionInd": "A",
```

```
                    "issuerCountryCode": "356",
                    "dsProtocolVersions": ["2.2.0", "2.3.1"],
                    "acsProtocolVersions": [
      {"version": "2.2.0",
      "acsInfoInd": ["01", "02"],
      "threeDSMethodURL": "https://www.acs.com/script1",
      "supportedMsgExt": [
            {"id": "A000000802-001","version": "2.0"},
            {"id": "A000000802-004","version": "1.0"}
      ]},
      {"version": "2.3.1",
      "acsInfoInd": ["01", "02", "03", "04", "81"],
      "threeDSMethodURL": "https://www.acs.com/script3"
      }
                        ]
                    }
                ]
}
```

The DS URL List data element contains information returned in a PRes message to the 3DS Server from the specific DS that contains the list of URLs that the 3DS Server can use to communicate with a DS.

The 3DS Server replaces the previous DS URL List with the latest received in the PRes message. If the DS URL List is absent from the PRes message, the 3DS Server deletes all existing DS URLs.

Its JSON Data Type: Array of objects contains:

- the 3DS Server to DS URL

- the DS Country Code (optional)

The detailed data elements are outlined in Table A.7.

Table A.7: DS URL List

New table, not replicated in this document.

DS URL List Data Example

New content, not replicated in this document.

A.11.1: Supported Message Extension Data Element

New content, not replicated in this document.

Table A.8 Supported Message Extension

New table, not replicated in this document.

Supported Message Extension Data Example

New content, not replicated in this document.

~~A.6~~A.12 Message Extension Data

~~A.6.1~~A.12.1 Message Extension Attributes

~~A.6.2~~A.12.2 Identification

~~A.6.3~~A.12.3 Criticality

~~A.7~~A.13 3DS Requestor Risk Information

~~A.7.1~~A.13.1 Cardholder Account Information

Table ~~A.8~~A.10: Cardholder Account Information

Data Element/Field Name	Description	Length/Format/Values
Cardholder Account Change	Date converted into UTC that the Cardholder's account with the 3DS Requestor was last changed, including Billing or Shipping address, new payment account, or new user(s) added.	
Cardholder Account Date	Date converted into UTC that the Cardholder opened the account with the 3DS Requestor.	
Cardholder Account Password Change	Date converted into UTC that Cardholder's account with the 3DS Requestor had a password change or account reset.	
Cardholder Account Purchase Count	If the Cardholder Account Purchase Count reaches the value 999, it remains set at 999.	Values accepted: • 0–999
Cardholder Account Requestor ID Field Name: `chAccReqID`	The 3DS Requestor assigned account identifier of the transacting Cardholder. This identifier is a coded as the SHA-256 + Base64url of the account identifier for the 3DS Requestor and is provided as a String.	Length: Maximum 64 characters JSON Data Type: String
Number of Transactions Per Year	If the maximum value is reached, the Number of Transactions Per Year remains set at 999.	Values accepted: • 0–999
Payment Account Age	Date converted into UTC that the payment account was enrolled in the Cardholder's account with the 3DS Requestor.	

Data Element/Field Name	Description	Length/Format/Values
Shipping Address Usage	Date converted into UTC when the shipping address used for this transaction was first used with the 3DS Requestor.	

~~A.7.2~~A.13.2 Merchant Risk Indicator

Table ~~A.9~~A.11: Merchant Risk Indicator

Data Element/Field Name	Description	Length/Format/Values
Shipping Indicator		• 08 = Pick-up and go delivery • 09 = Locker delivery (or other automated pick-up)
Transaction Characteristics Field Name: `transChar`	Indicates to the ACS specific transactions identified by the Merchant.	Size: Variable, 1–2 elements JSON Data Type: Array of string String: 2 characters Value accepted: • 01 = Cryptocurrency transaction • 02 = NFT transaction

~~A.7.3~~A.13.3 3DS Requestor Authentication Information

The 3DS Requestor Authentication Information contains optional information about how the cardholder authenticated during login to their 3DS Requestor account. The 3DS Requestor Authentication Information format is an array of object, the object contains the optional ~~The detailed~~ data elements, ~~which are optional, are~~as outlined in Table ~~A.10~~A.12.

Table ~~A.10~~A.12: 3DS Requestor Authentication Information

Data Element/Field Name	Source	Description	Length/Format/Values
3DS Requestor Authentication Data	3DS Server	For example, if the 3DS Requestor Authentication Method is: • 06, then this element can carry the FIDO Assertion and/or Attestation Data ~~(including the signature)~~. • 07, then this element can carry FIDO Assertion and/or Attestation with the FIDO Assurance Data signed by a trusted third party. For 3DS Requestor Authentication Method = 06 or 07, refer to the *EMV® 3-D Secure White Paper – Use of FIDO® Data in 3-D Secure Messages* for the 3DS Requestor Authentication Data content and format.	Length: Variable, m~~M~~aximum ~~20000~~50000 characters JSON Data Type: String or Object
3DS Requestor Authentication Method	3DS Server	Note: For 09 = SPC Authentication, the Assertion Data is provided as a JSON object returned by the SPC API.	• 09 = SPC Authentication • 10 = Electronic ID Authentication Data • 11–79 = Reserved for EMVCo future use (values invalid until defined by EMVCo)
3DS Requestor Authentication Timestamp	3DS Server	Date and time ~~in UTC~~ of the cardholder authentication ~~in the~~converted into UTC ~~Time Reference.~~	

Data Element/Field Name	Source	Description	Length/Format/Values
DS Authentication Information Verification Indicator Field Name: dsAuthInfVerifInd	DS	Value that represents the signature verification performed by the DS on the mechanism (e.g. FIDO) used by the Cardholder to authenticate to the 3DS Requestor. The DS populates this data element prior to passing to the ACS.	Length: 2 characters JSON Data Type: String Values accepted: • 01 = Verified • 02 = Failed • 03 = Not performed • 04–79 = Reserved for EMVCo future use (values invalid until defined by EMVCo) • 80–99 = Reserved for DS use

~~A.7.4~~A.13.4 3DS Requestor Prior Transaction Authentication Information

The 3DS Requestor Prior Transaction Authentication Information contains optional information about a 3DS cardholder authentication that occurred prior to the current transaction. The 3DS Requestor Prior Authentication Information format is an array of object, the object contains the optional data elements as ~~The detailed data elements, which are optional, are~~ outlined in Table ~~A.11~~A.13.

Table A.13: 3DS Requestor Prior Transaction Authentication Information

Data Element/Field Name	Description	Length/Format/Values
3DS Requestor Prior DS Transaction ID Field Name: `threeDSReqPriorDsTransId`	This data element provides the prior DS Transaction ID to the ACS to determine the best approach for handling a request.	Length: 36 characters JSON Data Type: String Value accepted: • This data element contains a DS Transaction ID for a prior authenticated transaction (for example, the first recurring transaction that was authenticated with the Cardholder).
3DS Requestor Prior Transaction Authentication Data		Length: maximum ~~2048~~20000 characters
3DS Requestor Prior Transaction Authentication Method		• 05 = SPC authentication • 06–79 = Reserved for EMVCo future use (values invalid until defined by EMVCo)
3DS Requestor Prior Transaction Authentication Timestamp	Date and time converted into UTC of the prior Cardholder authentication.	

A.13.5 ACS Rendering Type

The ACS Rendering Type ~~contains~~ identifies required elements and provides information about the rendering type that the ACS is sending for the cardholder authentication. The detailed data elements are outlined in Table ~~A.12~~A.14.

Table A.14: ACS Rendering Type

Data Element/Field Name	Description	Length/Format/Values	Inclusion
ACS Interface			R
ACS UI Template	Valid values for each Interface: Native UI = 01–04, ~~06~~07 HTML UI = 01–~~0506~~07 Note: HTML Other and HTML OOB are only valid in combination with 02 = HTML UI. If used with 01 = Native UI, the DS will respond with Error = 203 as described in Sections 5.9.3 and 5.9.8.	Length: 2 characters JSON Data Type: String Values accepted: • 06 = HTML OOB • 07 = Information	R
Device User Interface Mode Field Name: `deviceUserInterfaceMode`	Indicates the user interface mode the ACS will present to the Cardholder for a challenge.	Length: 2 numeric characters JSON Data Type: String Values accepted: • 01 = Portrait • 02 = Landscape • 03 = Voice • 04 = Other	R

JSON Object Example

```
{
"acsRenderingType":{ "acsInterface":"02","acsUiTemplate":03","deviceUserInterfaceMode":"02" }
 }
```

A.13.6 Device Rendering Options Supported

The detailed data elements are outlined in Table ~~A.13~~A.15.

Table ~~A.13~~A.15: Device Rendering Options Supported

Data Element/Field Name	Description	Length/Format/Values	Inclusion
SDK Authentication Type Field Name: `sdkAuthenticationType`	Authentication methods preferred/supported by the SDK in order of preference.	Size: 1–99 elements JSON Data Type: Array of string String: 2 characters • 01 = Static Passcode • 02 = SMS OTP • 03 = Key fob or EMV card reader OTP • 04 = App OTP • 05 = OTP Other • 06 = KBA • 07 = OOB Biometrics • 08 = OOB Login • 09 = OOB Other • 10 = Other	O

Data Element/Field Name	Description	Length/Format/Values	Inclusion
		• 11 = Push Confirmation • 12–79 = Reserved for EMVCo future use (values invalid until defined by EMVCo) • 80–99 = Reserved for DS use	
SDK Interface			R
SDK UI Type	Valid values for each Interface: Native UI = 01–04, and 07 HTML UI = 01–~~05~~07	~~Length: 2 characters~~ Size: 1–7 elements JSON Data Type: Array of string String: 2 characters ~~String v~~Values accepted: • 06 = HTML OOB (valid only for HTML UI) • 07 = Information	R

JSON Object Example:

```
{
"deviceRenderOptions":{ "sdkInterface":"03", "~~sdkUiType":[~~"sdkAuthenticationType":["01", "02", "03", "04",
"05"~~]~~, "06", "07", "08", "09", "10", "11", "12"], "sdkUiType":["01", "02", "03", "04", "05", "06", "07"] }
 }
```

A.13.7 Challenge Data Entry

~~Table A.14~~A.16 identifies the 3-D Secure message handling when this element is missing, assuming that no other errors are found. For ACS UI Type = 01, the Challenge Data Entry is considered missing in the table when both the Challenge Data Entry (`challengeDataEntry`) and the optional Challenge Data Entry 2 (`challengeDataEntryTwo`) are missing.

Table ~~A.14~~A.16 Challenge Data Entry

Challenge Data Entry	ACS UI Type	Challenge Cancelation Indicator	Resend Challenge Information Code	Challenge Additional Code	Challenge No Entry	Response
Missing	01, 02, or 03	Missing	Missing	Missing	Present • Value = Y	The ACS assumes that the Cardholder has not entered challenge data in the UI and therefore the ACS does not send the 3DS SDK an Error Message, but instead sends a CRes message.
Missing	01, 02, or 03	Present	Missing	Missing	Missing	The ACS sends the 3DS SDK a CRes message.
Missing	01, 02 or 03	Missing	Present • Value = Y	Missing	Missing	The ACS sends the 3DS SDK a CRes message.
~~Missing~~	~~01, 02, or 03~~	~~Missing~~	~~Present~~ ~~• Value = N~~		~~Present~~ ~~• Value = Y~~	~~The ACS assumes that the Cardholder has not entered challenge data in the UI and therefore the ACS does not send the 3DS SDK an Error Message, but instead sends a CRes message.~~
Missing	01, 02, or 03	Missing	Missing	Present Value = Y	Missing	The ACS sends the 3DS SDK a CRes message.
~~Missing~~	~~01, 02, or 03~~	~~Missing~~	~~Present~~ ~~• Value = N~~		~~Present~~ ~~• Value = Y~~	~~The ACS assumes that the Cardholder has not entered challenge data in the UI and therefore the ACS does not send the 3DS SDK an Error Message, but instead sends a CRes message.~~

Challenge Data Entry	ACS UI Type	Challenge Cancelation Indicator	Resend Challenge Information Code	Challenge Additional Code	Challenge No Entry	Response
Missing	01, 02, or 03	Present	Present	Missing	Present • Value = Y	If at least two of the fields Challenge Cancelation Indicator, Resend Challenge Information Code and/or Challenge No Data Entry are present, the ACS sends the 3DS SDK an Error Message.

The following Note was added directly below Table A.16:

Note: For all the combinations of Challenge Data Entry, Challenge Cancelation Indicator, Resend Challenge Information Code, Challenge Additional Code and Challenge No Entry not present in Table A.16, the ACS sends an Error Message with Error Code = 203 to the 3DS SDK.

~~A.7.8~~A.13.8 Transaction Status Conditions

The conditions on which indicators are valid within the 3-D Secure messages are outlined in Table ~~A.15~~A.17.

Table ~~A.15~~A.17: Transaction Status Conditions

Transaction Status ~~Indicator~~	ARes	Final CRes	RReq	Error Response
C = Challenge Required; Additional authentication is required using the CReq/CRes	Valid~~11~~9			
D = Challenge Required; Decoupled Authentication confirmed	Valid10	~~Invalid~~Valid11	~~Invalid~~Valid11	• Final CRes: ~~End processing (no Error)~~ Not applicable • RReq: Refer to Section 5.9.8 and use Error Code = 203 if Condition not met
S = Challenge using SPC	Valid13	Invalid	Invalid	• ARes: Refer to Section 5.9.3 and use Error Code = 203 if Condition not met

Transaction Status ~~Indicator~~	ARes	Final CRes	RReq	Error Response
				• Final CRes: End processing (no Error) • RReq: Refer to Section 5.9.8 and use Error Code = 203

Footnote 9: This indicator (C) is not valid if Device Channel = 03, or if the 3DS Requestor Challenge Indicator = 06 (No challenge requested; Data share only) within the AReq message.

Footnote 10: This indicator (D) can be sent only if 3DS Requestor Decoupled Request Indicator = Y or B within the AReq message.

Footnote 11: This indicator (D) can be sent only if 3DS Requestor Decoupled Request Indicator = F or B within the AReq message.

Footnote 13: This indicator (S) can be sent only if 3DS Requestor SPC Support = Y within the AReq message.

A.13.9 Multi-Transaction

New section and Table A.18. Content is not replicated in this document.

A.13.10 Seller Information

New section and Table A.19. Content is not replicated in this document.

~~A.8~~A.14 UI Data Elements

Table ~~A.18~~A.20 specifies the placement and the ~~presence~~inclusion of UI data elements on the UI with respect to the zones defined in Section 4.1.

- M = Mandatory ~~presence~~inclusion
- O = Optional ~~presence~~ —Optional to provide for the ACS; If present, Mandatory to display for the SDK
- N = Not present

Table ~~A.18~~A.20 UI Data Elements

Table A.20 was revised and updated. It is not replicated in this document.

Note: The data elements listed in Table A.20 are not needed for ACS UI Type = 05 and 06 (HTML and HTML OOB template).

A.14.1 Issuer Image

The Issuer Image (`issuerImage`) is supplied by the ACS to be displayed during the challenge message exchange. The detailed format of this data element is provided in Table ~~A.16~~A.21.

Depending on the display capabilities and mode set by the Cardholder, the 3DS SDK uses:

- The Default Image if it is the only image provided by the ACS, OR if dark mode is not enabled on the device, OR the device display is not monochrome-only.
- The Dark Mode Image if the 3DS SDK detects that dark mode is enabled on the device.
- The Monochrome Image if the SDK is running on a device that only supports monochrome display.

Table ~~A.16~~A.21 Issuer Image

Data Element/Field Name	Description	Length/Format/Values

• ~~Medium Density~~Default Image
Field Name: ~~`medium`~~`default`
• ~~High Density~~Dark Mode Image
Field Name: ~~`high`~~`dark`
• ~~Extra High Density~~Monochrome Image
Field Name: ~~`extraHigh`~~`monochrome`
	~~Include up to three fully qualified URLs defined as either; medium density, high density and extra high-density images of the Issuer Image.~~
	Include at minimum one and at maximum three Fully Qualified URLs defined as default, dark mode or monochrome images of the Issuer Image.
	Examples:
	~~Images to display:~~
	`"issuerImage" :{`
	`"`~~`medium`~~`default": "https://acs.com/`~~`medium`~~`default_image.png",`
	`"`~~`high`~~`dark": "https://acs.com/`~~`high`~~`dark_image.png",`
	`"`~~`extraHigh`~~`monochrome": "https://acs.com/`~~`extraHigh`~~`monochrome_image.png"`
	`}`
		Length: Variable, maximum ~~2048~~6144 characters
		JSON Data Type: ~~String~~JSON object
		Values accepted:
		~~Images to display:~~
		• Fully Qualified URL in correct JSON object format
		If present, the Issuer Image object shall contain, at minimum, the Default Image.

~~A.13.10~~A.14.2 Payment System Image

The Payment System Image (psImage) is supplied by the ACS to be displayed during the challenge message exchange. The detailed format of the data element is provided in Table ~~A.17~~A.22.

Depending on the display capabilities and mode set by the Cardholder, the 3DS SDK uses:

- The Default Image if it is the only image provided by the ACS, OR if dark mode is not enabled on the device, OR if the device display is not monochrome-only.
- The Dark Mode Image if the 3DS SDK detects that dark mode is enabled on the device.
- The Monochrome Image if the 3DS SDK is running on a device that only supports monochrome display.

Table ~~A.17~~A.22 Payment System Image

Data Element/Field Name	Description	Length/Format/Values

• ~~Medium Density~~Default Image
Field Name: ~~medium~~default
• ~~High Density~~Dark Mode Image
Field Name: ~~high~~dark
• ~~Extra High Density~~Monochrome Image
Field Name: ~~extraHigh~~monochrome
	~~Include up to three fully qualified URLs defined as either; medium density, high density and extra-high density images of the DS or Payment System image.~~
	Include at minimum one and at maximum three Fully Qualified URLs defined as default, dark mode or monochrome images of the DS or Payment System image.
	Examples:
	~~Images to display:~~
	"psImage" :{
	"~~medium~~default": "https://acs.com/~~medium~~default_image.png",
	"~~high~~dark": "https://acs.com/~~high~~dark_image.png",
	"~~extraHigh~~monochrome": "https://acs.com/~~extraHigh~~monochrome_image.png"
	}
		Length: Variable, maximum ~~2048~~6144
		JSON Data Type: ~~String~~JSON Object
		Value accepted:
		Images to display:
		• Fully Qualified URL in correct JSON object format
		If present, the Payment System Image object shall contain, at minimum, the Default Image.

A.15 iframe and Sandbox Attributes

Section A.15 and (including Tables A.23 and A.24) is a new section and is not replicated in this specification bulletin.

A.16 3-D Secure Array Fields

Section A.16 is a new section and is not replicated in this specification bulletin.

A.17 EMV Payment Token Information

Section A.17 (including Table A.25) is a new section and is not replicated in this specification bulletin.

A.18 Challenge Text Box Settings

Section A.18 (including Table A.26) is a new section and is not replicated in this specification bulletin.

A.19 Broadcast Information

Section A.19 (including Table A.27) is a new section and is not replicated in this specification bulletin.

A.20 Cardholder Information Text

Section A.20 (including Figures A.1 and A.2) is a new section and is not replicated in this specification bulletin.

A.21 SPC Transaction Data

Section A.21 (including Table A.28) is a new section and is not replicated in this specification bulletin.

A.22 HTTP Headers

Section A.22 (including Table A.29 and HTTP Header Examples) is a new section and is not replicated in this specification bulletin.

Annex B Message Format

B.1 AReq Message Data Elements

Table B.1 AReq Data Elements

Data Element	Field Name
3DS Method ID	threeDSMethodId
~~3DS Requestor Authentication Method Verification Indicator~~	~~threeDSReqAuthMethodInd~~
3DS Requestor SPC Support	threeDSRequestorSpcSupport
Accept Language	acceptLanguage
Acquirer Country Code	acquirerCountryCode
Acquirer Country Code Source	acquirerCountryCodeSource
App IP Address	appIp
Browser User Device ID	deviceId
Browser User ID	userId
Card Security Code	cardSecurityCode
Card Security Code Status	cardSecurityCodeStatus
Card Security Code Status Source	cardSecurityCodeStatusSource
Default-SDK Type	defaultSdkType
Device Binding Status	deviceBindingStatus
Device Binding Status Source	deviceBindingStatusSource
EMV Payment Token Information	payTokenInfo
Multi-Transaction	multiTransaction
Payee Origin	payeeOrigin
Recurring Amount	recurringAmount
Recurring Currency	recurringCurrency
Recurring Currency Exponent	recurringExponent

Recurring Date	recurringDate
Recurring Indicator	recurringInd
SDK Server Signed Content	sdkServerSignedContent
SDK Type	sdkType
Seller Information	sellerInfo
SPC Incompletion Indicator	spcIncompInd
Split-SDK Type	splitSdkType
Tax ID	taxId
Trust List Status	trustListStatus
Trust List Status Source	trustListStatusSource
~~Whitelist Status~~	~~whiteListStatus~~
~~Whitelist Status Source~~	~~whiteListStatusSource~~

B.2 ARes Message Data Elements

Table B.2 ARes Data Elements

Data Element	Field Name
3DS Requestor App URL Indicator	threeDSRequestorAppURLInd
Authentication Method	authenticationMethod
~~Authentication Type~~	~~authenticationType~~
Card Security Code Status	cardSecurityCodeStatus
Card Security Code Status Source	cardSecurityCodeStatusSource
Device Binding Status	deviceBindingStatus
Device Binding Status Source	deviceBindingStatusSource
Device Information Recognised Version	deviceInfoRecognisedVersion
SPC Transaction Data	spcTransData
Transaction Challenge Exemption	transChallengeExemption
Transaction Status Reason Information	transStatusReasonInfo
Trust List Status	trustListStatus
Trust List Status Source	trustListStatusSource
WebAuthn Credential List	webAuthnCredList
~~Whitelist Status~~	~~whiteListStatus~~
~~Whitelist Status Source~~	~~whiteListStatusSource~~

B.3 CReq Message Data Elements

Table B.3 CReq Data Elements

Data Element	Field Name
Challenge Additional Code	challengeAddCode
Challenge Data Entry 2	challengeDataEntryTwo
Device Binding Data Entry	deviceBindingDataEntry
Information Continuation Indicator	infoContinueIndicator
OOB App Status	oobAppStatus
OOB App URL Indicator	oobAppURLInd
Trust List Data Entry	trustListDataEntry
~~Whitelisting Data Entry~~	~~whiteListingDataEntry~~

B.4 CRes Message Data Elements

Table B.4 CRes Data Elements

Data Element	Field Name
Challenge Additional Label	challengeAddLabel
Challenge Entry Box	challengeEntryBox
Challenge Entry Box 2	challengeEntryBoxTwo
Device Binding Information Text	deviceBindingInfoText
Information Continuation Label	infoContinueLabel
Toggle Position Indicator	togglePositionInd
Trust List Information Text	trustListInfoText
~~Whitelisting Information Text~~	~~whiteListInformationText~~

B.6 PReq Message Data Elements

Table B.6: PReq Data Elements

Data Element	Field Name
Card Range Data Download Indicator	cardRangeDataDownloadInd

B.7 PRes Message Data Elements

Table B.7 PRes Data Elements

Data Element	Field Name
Card Range Data File URL	cardRangeDataFileURL
~~DS End Protocol Version~~	~~dsEndProtocolVersion~~
DS Protocol Versions	dsProtocolVersions
~~DS Start Protocol Version~~	~~dsStartProtocolVersion~~
DS URL List	dsUrlList
Read Order	readOrder

B.8 RReq Message Data Elements

Table B.8 RReq Data Elements

Data Element	Field Name
Authentication Method	authenticationMethod
~~Authentication Type~~	~~authenticationType~~
Cardholder Information Text	cardholderInfo
Challenge Error Reporting	challengeErrorReporting
Device Binding Status	deviceBindingStatus
Device Binding Status Source	deviceBindingStatusSource
Transaction Status Reason Information	transStatusReasonInfo
Trust List Status	trustListStatus
Trust List Status Source	trustListStatusSource
~~Whitelist Status~~	~~whiteListStatus~~
~~Whitelist Status Source~~	~~whiteListStatusSource~~

B.10 OReq Message Data Elements

New section and Table B.10. Content is not replicated in this document.

B.11 ORes Message Data Elements

New section and Table B.11. Content is not replicated in this document.

Legal Notice

The EMV® Specifications are provided “AS IS” without warranties of any kind, and EMVCo neither assumes nor accepts any liability for any errors or omissions contained in these Specifications. EMVCO DISCLAIMS ALL REPRESENTATIONS AND WARRANTIES, EXPRESS OR IMPLIED, INCLUDING WITHOUT LIMITATION IMPLIED WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE, TITLE AND NON-INFRINGEMENT, AS TO THESE SPECIFICATIONS.

EMVCo makes no representations or warranties with respect to intellectual property rights of any third parties in or in relation to the Specifications. EMVCo undertakes no responsibility to determine whether any implementation of the EMV® Specifications may violate, infringe, or otherwise exercise the patent, copyright, trademark, trade secret, know-how, or other intellectual property rights of third parties, and thus any person who implements any part of the EMV® Specifications should consult an intellectual property attorney before any such implementation.

Without limiting the foregoing, the Specifications may provide for the use of public key encryption and other technology, which may be the subject matter of patents in several countries. Any party seeking to implement these Specifications is solely responsible for determining whether its activities require a license to any such technology, including for patents on public key encryption technology. EMVCo shall not be liable under any theory for any party’s infringement of any intellectual property rights in connection with the EMV® Specifications