Test Harness Specification

UL 3DS Self Test Platform

Author	UL
Version	1.2.1
Date	May 23, 2019
Status	Final
Classification	Confidential

UL
E-mail transactionsecurity@ul.com
Website www.ul.com/transactionsecurity

Version history

Version	Date	Status	Author
1.0.0	**08-May-2017**	**Draft**	**UL**
1.0.5	**05-March-2018**	**Draft**	**UL**
1.0.6	**26-March-2018**	**Draft**	**UL**
1.0.7	**09-May-2018**	**Final**	**UL**
1.0.8	**29-May-2018**	**Draft**	**UL**
1.0.9	**15-Jun-2018**	**Draft**	**UL**
1.1.0	**21-Aug-2018**	**Final**	**UL**
1.1.1	**07-Nov-2018**	**Final**	**UL**
1.1.2	**29-Nov-2018**	**Final**	**UL**
1.2.0	**01-May-2019**	**Final**	**UL**
1.2.1	**23-May-2019**	**Final**	**UL**

Change history

Version	Date	Changes
1.0.7	**09-May-2018**	• Updated eci acceptable values (removed the unhappy flow values due to removal of test cases) • Corrected the Frictionless configuration data • Decoupled p_messageVersion value from the Test Harness version value
1.0.8	**29-May-2018**	• Removed the reference that a DS can use their own Root CA for the platform. This feature will be available in the future, but was removed to avoid confusion at the moment. • Removed 'threeDSServerRefNumber' data element from the pArs • Changed to 'https' the value for notificationURL in the pArq message example (notificationURL: "https://exampleofnotificationurl.com") • Changed acschallengeMandated to have the correct capitalization of acsChallengeMandated (as per EMVco specification) • Added note (under A.2.1 Table) to clarify question on market or regional restriction on conditional data elements: "Note: Since market or region restriction may vary, any conditional data element with a market or region restriction should be treated as optional for testing purposes." • Updated acceptable values of eci (Section 2.8) • Added a note for pIrq/pIrs exchange for Application OOB flow (Section 3.3)

1.0.9	**15-Jun-2018**	• Added new Account Range #27 to trigger Challenge flow providing some freedom to the ACS with respect to the used ACS UI Type. • Corrected a typo in the configuration profile data: nbPurchaceAccount should have been nbPurchaseAccount. • Corrected the format of p_isTransactionCompleted used in the pSrq message from boolean to string.
1.1.0	**21-Aug-2018**	• Added 'SDK Challenge' in Table 5. Challenge Type listeners • Changed minimum required API Level for the UL Reference Application on Android from 19 to 21 • Removed quotes from value of criticalityIndicator in the example pPrq message as the field has type JSON boolean. • Corrected Base 64 encoding to Base64url encoding in Annex A.5.2 to be in line with the EMV 3-D Secure Protocol and Core Functions specification. • Added more realistic examples to Annex A.5.2 for the CReq/CRes HTTP POST in case of the 02-BRW Device Channel. • Removed the note below Table 6 (Card ranges) regarding the order in which the card ranges should be included in the cardRangeData data element as that is no longer required to pass the relevant test cases. • Added new note below Table 6 regarding the inclusion of Card Range #19 and #20 in the PRes messages sent by a DS. • Changed 'threeDSServerTransactionID' to 'threeDSServerTransID' in section A.2.9 • Added 'challengeWindowSize' as mandatory field in pGcq A.2.3.
1.1.1	**07-Nov-2018**	• Added more detailed information regarding p_formValues_BRW and p_formValues_APP data elements used in the pIrq message. • Added Annex A.7 describing proprietary Error Component values.
1.1.2	**29-Nov-2018**	• Update description about certificates and Root CA • Added explanation for SDK Product Providers regarding the preferred screen resolution provided by the device used for testing.

1.2.0	**01-May-2019**	• Changed description of Account Number Range 27 to indicate that any Native UI can be used by the ACS to perform the challenge. • Added new Account Number Range 28 to perform a challenge flow using an HTML UI (in case of 01-APP Device Channel). • Added section 1.5 regarding supported 3-D Secure Protocol Version Numbers. • Added data elements introduced in EMV® 3-D Secure Protocol and Core Functions specification, version 2.2.0 (December 2018) to the applicable proprietary messages. • Added new section to describe the Decoupled Authentication flow in case the ACS is the SUT. • Added new Account Number Ranges 29 to 32 to allow testing v2.2.0 features. • Updated the notes below the Account Number Ranges table, clarifying the use of Protocol Version number related elements in the Card Range Data as well as the usage of ACS Information Indicator. • Section A.1.3 “Configuration Data Profile Standard Values” Removed • Section A.4 “SDK Challenge Listener Example Code” has been removed. References to Section A.4 have been replaced with references to the iOS and Android 3DS Reference App Developers Guides. • Drop data element ‘threeDSReqAuthMethodInd’ from Section A.2.1 Proprietary Authentication Request • Update presence of ‘threeDSRequestorDecMaxTime’ data element to ‘C’ in Section A.2.1 Proprietary Authentication Request • Clarified usage of serialNum data element in first PReq message of a test case. • Clarified usage of threeDSServerTransID and threeDSRequestorURL data elements in pPrq messages.
1.2.1	**23-May-2019**	• Removed spurious line breaks from the DS Public Keys listed in Annex A.5.

Table Of Contents

1 Introduction

To integrate a 3-D Secure component with the UL 3DS Self Test Platform (UL 3DS STP) certain functionalities outside the scope of the EMV® 3-D Secure specifications[1] are required prior to testing. These required functionalities form the test harness. This test harness allows for the proper connection with the Test Platform as required to run tests, perform certification and handle functionality outside the scope of the EMV® 3-D Secure specifications[1] (for example Out-Of-Band processing). The required functionality depends on the System Under Test (SUT) that is connected to the Test Platform.

The functionality described in this document is required to be implemented on the Product Provider side before testing with the platform can start.

This document provides an overview of the functionality that is expected for each SUT, as well as the contents and formatting of proprietary messages that are being exchanged between the SUT and the Test Platform. Furthermore, examples of these messages and the configuration data that is used during testing are made available.

1.1 Definitions

1.1.1 Test Data

Test Data is used by the simulators to generate the request and response messages defined in the test cases. This is data that the platform will use to populate messages and must be formatted according to the EMV 3-D Secure specifications[1].

1.1.2 Configuration Data

Configuration Data links to the Test Data and is used to trigger certain behavior in the SUT. Profiles are defined for each expected behavior and linked to a BIN-range that is used as a trigger.

1.2 Abbreviations

Table 1 summarizes the abbreviations used throughout this document.

Table 1 Abbreviations

Abbreviation	Definition
3DS	3-D Secure
ACS	Access Control Server
DS	Directory Server
SDK	Software Development Kit
STP	Self Test Platform
SUT	System under Test
3RI	3DS Requestor Initiated
OOB	Out-of-Band
URL	Uniform Resource Locator (standard http:// or https:// link)
	UL proprietary messages
pArq	proprietary Authentication Request
pArs	proprietary Authentication Response
pIrq	proprietary Information Request
pIrs	proprietary Information Response
pOrq	proprietary Out-of-Band Request
pOrs	proprietary Out-of-Band Response
pPrq	proprietary Preparation Request
pPrs	proprietary Preparation Response
pGcq	proprietary Get Challenge Request
pGcs	proprietary Get Challenge Response
pSrq	proprietary SDK Information Request
pSrs	proprietary SDK Information Response

1.3 Proprietary Protocol Version Number

Proprietary Protocol Version Number	Status
1.0.0	Deprecated
1.0.5	Active

1.4 Ecosystem overview

The 3-D Secure ecosystem contains 4 systems that can be considered SUT. In three of these systems, the ACS (Section 3), SDK (Section 5) and 3DS Server (Section 6), an additional test harness is necessary to properly connect with the UL 3DS Self Test Platform. The DS (Section 4) does not require any additional test harness to connect with the UL 3DS Self Test Platform but will be expected to respond to transactions based on the incoming test data as described in this document. Figure 1 shows an overview of the required test harness for each system in a simplified overview of the 3DS ecosystem.

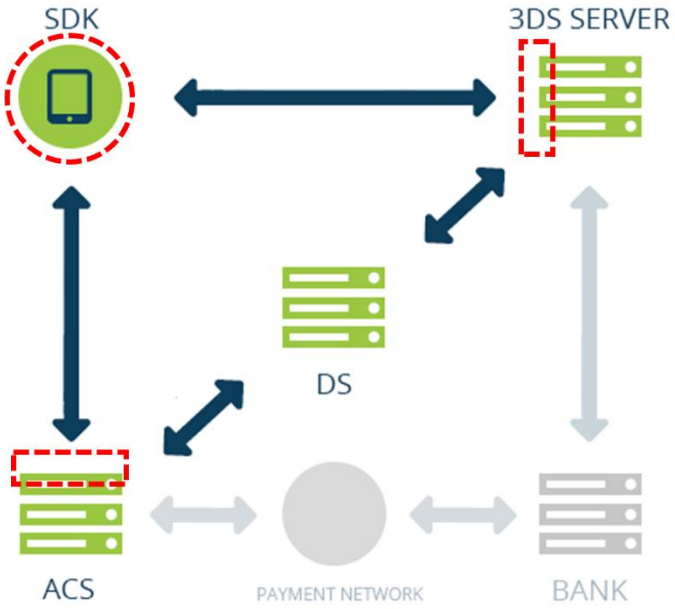


Figure 1 A simplified overview of the 3-D Secure ecosystem indicating the interfaces where the respective SUT will connect to the UL 3DS Self Test Platform.

For the 3DS Server and the ACS the test harness consists of the functionality and configuration data described in the rest of the documentation.

For the SDK there is an UL Reference Application available that can be used to connect to the Test Platform. This Reference Application will be made available via an appropriate channel and communicated to the Product Provider during setup. The reference application will take care of the majority of functionality needed to interact with the Test Platform. If automation of (most of the) SDK testing is preferred, additional logic needs to be added to the SDK to facilitate this. This is described in Section 5 and in the 3DS Reference App Developers Guide for iOS and Android.

In the EMV 3-D Secure specifications[1] there are three device channels distinguished (App-based, Browser-based and 3DS Requestor Initiated 3RI) that indicate from which channel was the transaction is originated. In the following subsections an overview of each device channel's proprietary messages is given. Through different Process Flow Overview designs (i.e. App-based flow, Browser-based flow, App-based flow Out-of-Band and 3DS Requestor Initiated based flow) the proprietary messages that are used by the UL 3DS Self Test Platform are distinguished and explained. Since the proprietary messages are involved in the process flow only when ACS or 3DS Server is the System Under Test, Sections 3 and 6 are devoted to explain in detail their functionality in each solution.

1.4.1 Application based flow - Overview

Figure 2 shows the flow for the App-based implementations including the steps where proprietary messages (in red color) need to be sent/received to/from the UL 3DS Self Test Platform.

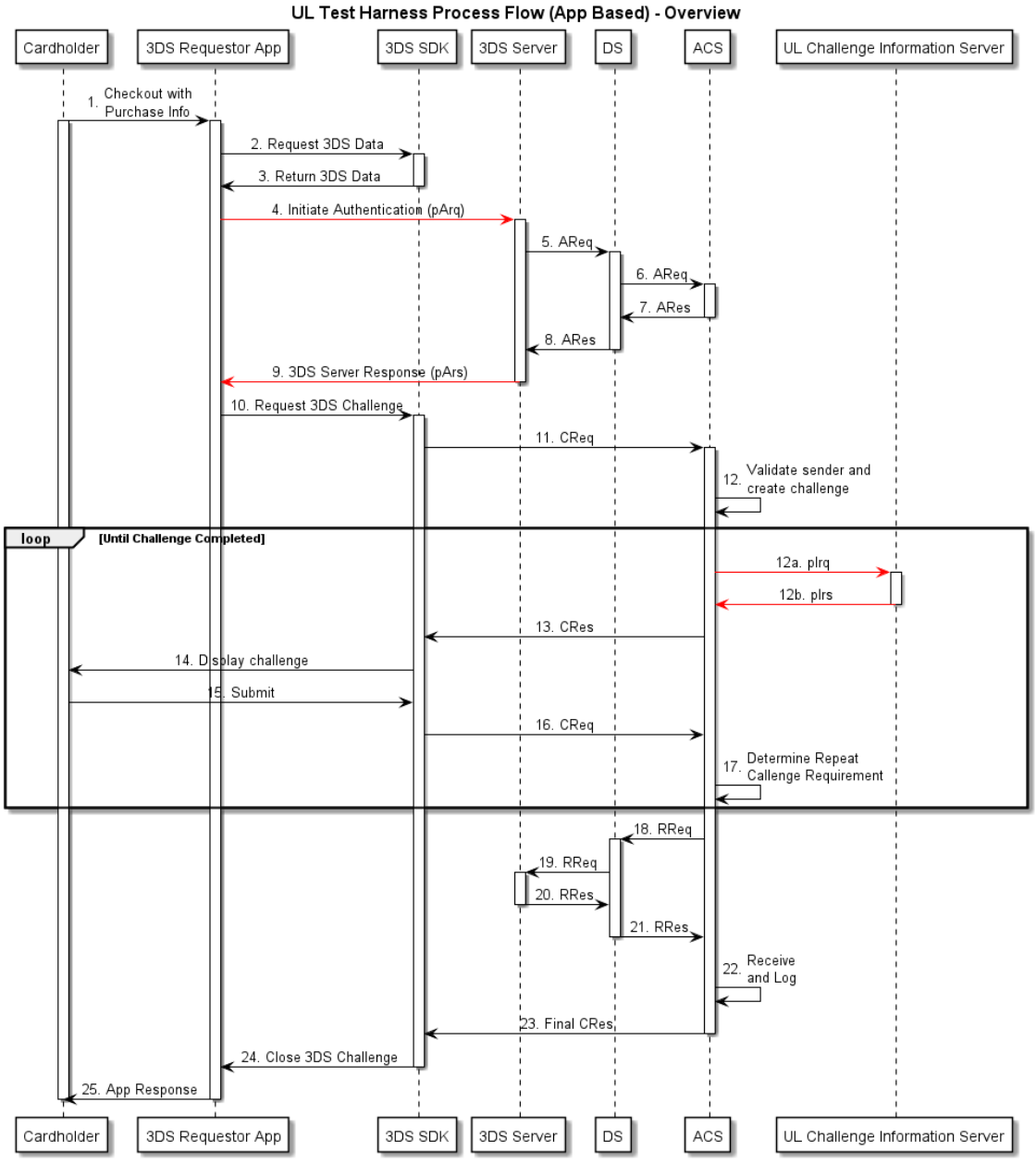


Figure 2 An overview of the 3DS 2.0 flow for the Application Device Channel including the steps where proprietary messages will be used.

As it can be noticed in Figure 2 there are two pairs of proprietary messages involved in this flow the proprietary Authentication Request (pArq), the proprietary Authentication Response (pArs), the proprietary Information Request (pIrq) and the proprietary Information Response (pIrs). These proprietary messages will be explained in detailed in Sections 3.1 and 6.1.

1.4.2 Browser based flow - Overview

Figure 3 shows the flow for the browser-based implementations including the steps where proprietary messages (in red color) need to be sent/received to/from the UL 3DS Self Test Platform.

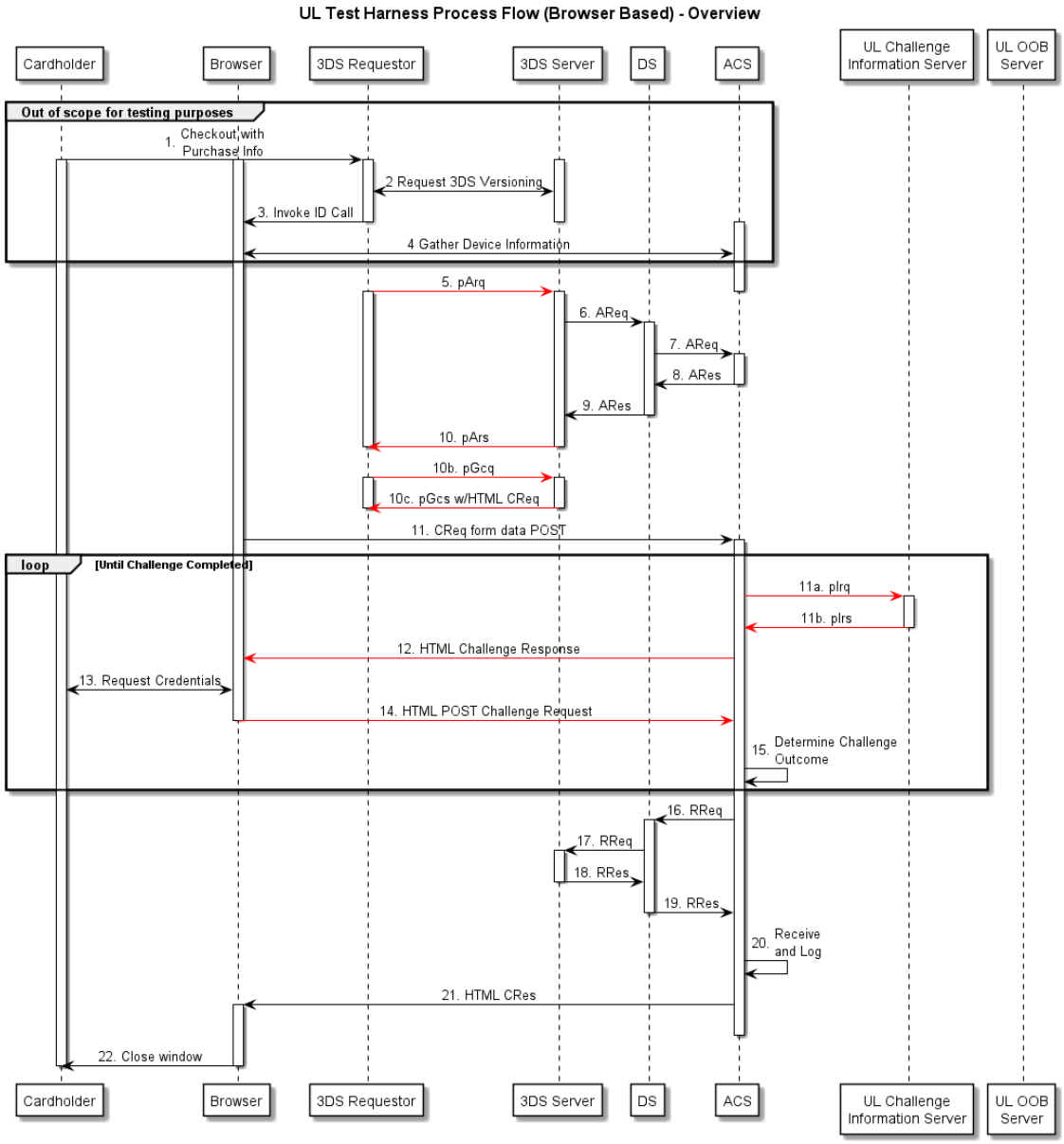


Figure 3 An overview of the 3DS 2.0 flow for the Browser Device Channel including the steps where proprietary messages will be used.

As can be seen in Figure 3 there are four pairs of proprietary messages involved in this flow: the proprietary Authentication Request (pArq), the proprietary Authentication Response (pArs), the proprietary Get challenge Request (pGcq), the proprietary Get challenge Response (pGcs), the HTML Challenge Response and Request, the proprietary Information Request (pIrq) and the proprietary Information Response (pIrs). Sections 3.2 and 6.2 explain these proprietary messages in detailed.

1.4.3 Application based flow Out-of-Band - Overview

Figure 4 presents the flow for the App-based Out-of-Band implementations including the steps where proprietary messages (in red color) need to be sent/received to/from the UL 3DS Self Test Platform.

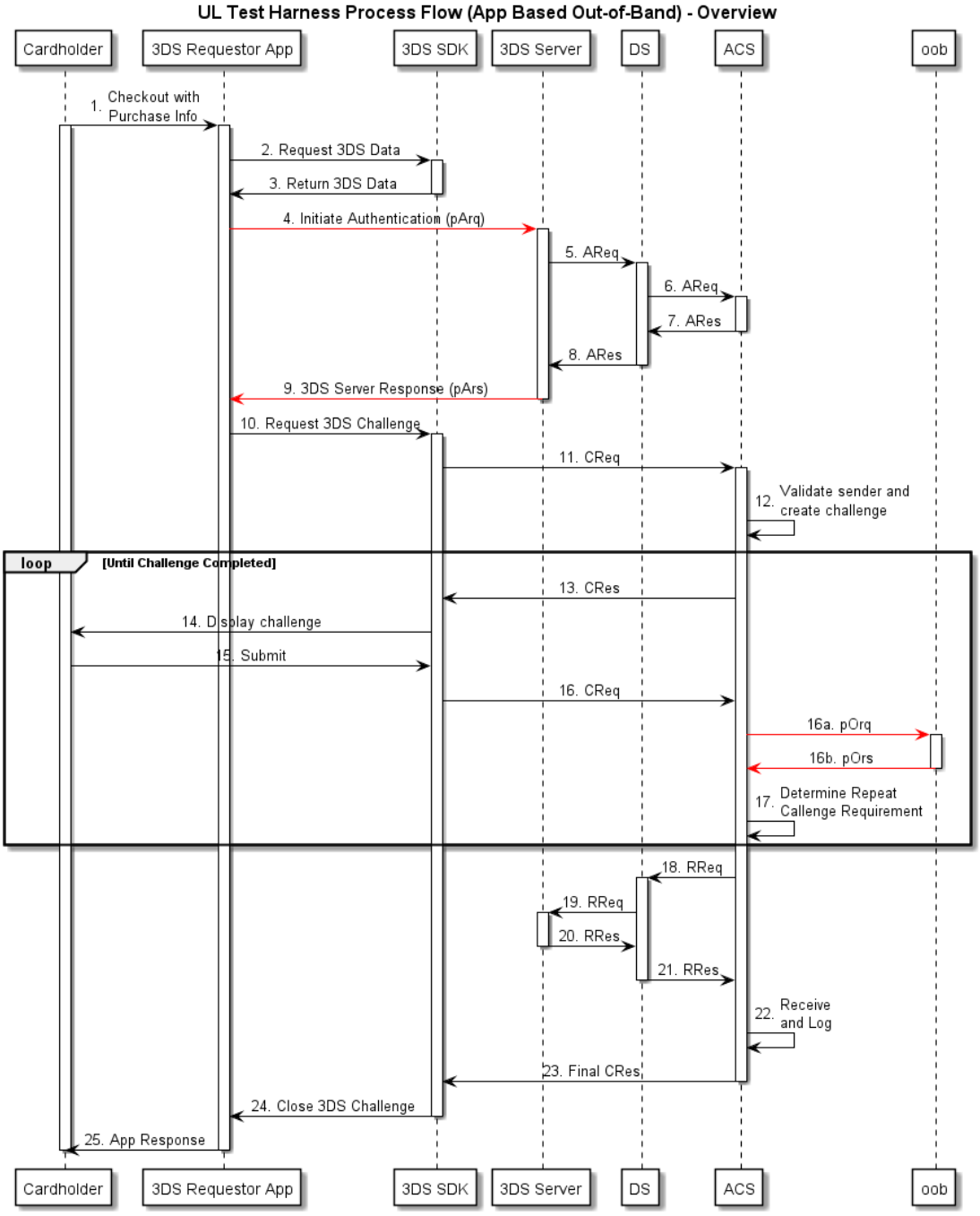


Figure 4 An overview of the 3DS 2.0 Out-of-Band flow for the Application Device Channel including the steps where proprietary messages will be used.

1.4.4 3DS Requestor Initiated based flow – Overview

Figure 5 shows the flow for the 3DS Requestor Initiated-based implementations including the steps where proprietary messages (in red color) need to be sent/received to/from the UL 3DS Self Test Platform.

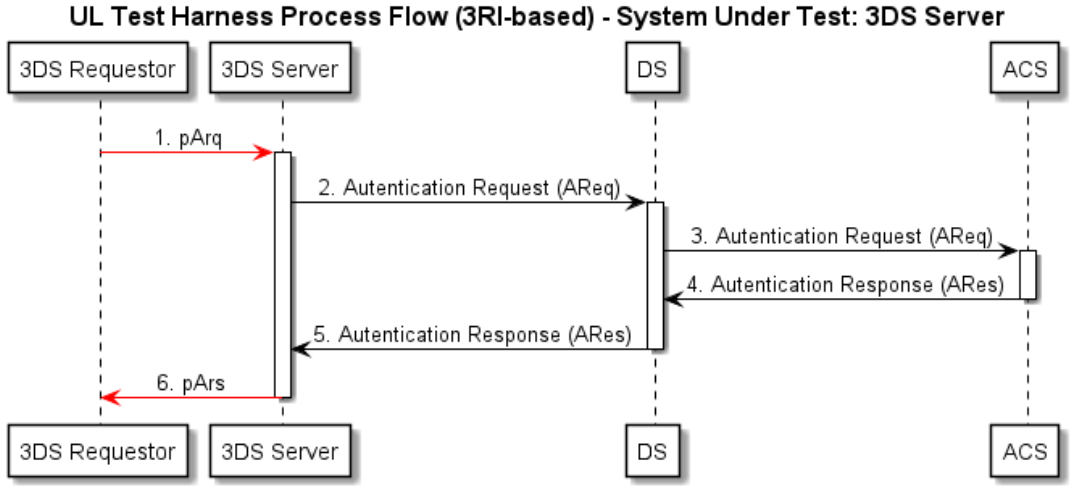


Figure 5 An overview of the 3DS 2.0 flow for the 3RI Device Channel including the steps where proprietary messages will be used.

As can be noticed in Figure 5, there is one pair of proprietary messages involved in this flow: the proprietary Authentication Request (pArq) and the proprietary Authentication Response (pArs). Section 6.4 explains these proprietary messages in detail.

1.5 Active Protocol Versions

This Test Harness document applies to both active 3-D Secure Protocol Version Numbers 2.1.0 and 2.2.0 with the following remarks:

- Systems Under Test only supporting 3-D Secure Protocol Version Number 2.1.0 should not include the data elements marked as 2.2.0 only in proprietary messages defined throughout this document.
- Systems Under Test only supporting 3-D Secure Protocol Version Number 2.1.0 do not need to support Account Number Ranges 29 to 32 defined in Annex A.1.
- Systems Under Test (more specifically Directory Servers) should list version 2.1.0 as Start Protocol Version Number and 2.2.0 as End Protocol Version Number in the relevant data elements included in PRes messages in case they support 3-D Secure Protocol Version Number 2.2.0. In case a DS only supports 3-D Secure Protocol Version Number 2.1.0, only that Protocol Version Number is mentioned in the PRes messages.
- A 3DS Server should use the same Message Version Number in the AReq message sent to the DS as received in the pArq message received from the 3DS Requestor Environment. In this way, the test case can drive the 3-D Secure Protocol Version Number used within the test case. Note, that the 3DS Test Platform in general will reject messages with Message Version Number 2.1.0 when executing test cases included in the v2.2.0 Test Plan and vice versa.

2 General requirements/values

This section describes the general requirements and values that are relevant for all systems under test.

2.1 Timeout values

In addition to the timeout values mentioned in the 3DS Specification the following timeout values must be considered and if applicable for your SUT be implemented.

2.1.1 ACS and 3DS Server as System Under Test

According to Requirement 228 [1], the 3DS Server and ACS shall set appropriate AReq message timeout values, after exceeding the connection is closed. To test the corresponding SUT behavior, a timeout value of 10 seconds is expected to be implemented for the 3DS Server or ACS.

2.1.2 DS as System Under Test

According to Requirement 234 [1], the DS shall set the ARes timeout value. The timeout values are specified by the DS, but since the functionality of configuring different timeout values is not yet present, a timeout value of 7 seconds is expected to be implemented.

2.2 Internal Timeout values for the Test System

When UL 3DS Self Test Platform acts as the receiving party of the error messages a timeout of 7 seconds will be set to respond to an incoming Erro request message. The 7 seconds will start from the ‘end’ of the test case, with the ‘end’ being defined as the point where the last message in a transaction has been received. The expected response will be a 200 OK message.

2.3 Proprietary message formatting and validation

The proprietary messages used during the testing will follow the same message formatting as the messages (AReq/ARes, RReq/RRes, PReq/PRes) specified in the EMVCo specifications [1] and will be validated according to the same rules. On an invalid request or response to a proprietary message, the UL Simulators may send back an EMVCo Erro Response code to the System Under test sending the invalid message.

The proprietary messages shall use the same Content-Type header as defined in Requirement 190[1] with the value : Content-Type: application/json;charset=UTF-8

2.4 HTTP Headers

The UL 3DS Self Test Platform will set the following headers on outgoing requests:

- x-ul-testcase-id identifies the Test Case Identifier of the running Test Case. This field can be used by a Product Provider to identify the source of a message.
- x-ul-testcaserun-id is the unique identifier of a single Test Case execution. This identifier will be the same for all requests sent for that execution.

These headers are only provided for convenience, and their name or formatting may change without prior notice.

Systems under test shall not set any of these headers.

2.5 Public keys

The platform requires that DS Product Providers configure their DS Public Key on the platform. The key can be in either EC or RSA format.

These public keys are prerequisite for achieving the mutual Transport Layer Security (TLS) authentication between the Product Provider's component and the UL 3DS Self Test Platform.

2.5.1 For a DS System Under Test

For a DS system under test for the purpose of mutual authentication, the Product Provider has the option to either use certificates generated with:

1. UL Root CA (default)
2. Product Provider Root CA

For more information on the process to obtain the certificates please go to the following page in the platform: Company Profile > Certificates

For a DS system under test for the purpose of signing the ACS content, the Product Provider has the option to either use certificates generated with:

1. UL Root CA (default)
2. Product Provider Root CA

For more information on the process to obtain and configure the certificates please go to the following page in the platform: Projects > Select Project ID > Configuration > Certificates

For a DS system under test for the purpose of encrypting the device information, the Product Provider has the option to either use certificates generated with:

1. UL Root CA (default)
2. Product Provider Root CA

For more information on the process to obtain and configure the certificates please go to the following page in the platform: Projects > Select Project ID > Configuration > Certificates

2.5.2 For an ACS or 3DS Server System Under Test

For an ACS or 3DS Server system under test, the Product Provider shall use a certificate generated with the UL Root CA.

For more information on the process to obtain the certificates please go to the following page in the platform: Company Profile > Certificates > UL

2.5.3 For an SDK System Under Test

For an SDK system under test, the UL DS Public Keys listed in A.5 can be used.

2.6 UL Test Reference Numbers

2.6.1 Reference numbers

To integrate with the UL 3DS Self Test Platform, systems under test must configure a reference number from this list for each component. A system must also recognize these reference numbers as participating reference numbers. Any other reference numbers (including EMVCo-assigned reference numbers) will be rejected.

Data Element / Field Name	Value
sdkReferenceNumber	3DS_LOA_SDK_PPFU_020100_00007 3DS_LOA_SDK_PPFU_020100_00011 3DS_LOA_SDK_PPFU_020100_00015

Data Element / Field Name	Value
threeDSServerRefNumber	3DS_LOA_SER_PPFU_020100_00008 3DS_LOA_SER_PPFU_020100_00012 3DS_LOA_SER_PPFU_020100_00016

Data Element / Field Name	Value
acsReferenceNumber	3DS_LOA_ACS_PPFU_020100_00009 3DS_LOA_ACS_PPFU_020100_00013 3DS_LOA_ACS_PPFU_020100_00017

Data Element / Field Name	Value
dsReferenceNumber	3DS_LOA_DIS_PPFU_020100_00010 3DS_LOA_DIS_PPFU_020100_00014 3DS_LOA_DIS_PPFU_020100_00018

Note: Any other reference number is to be treated as non-participating (SDK/3DS Server/DS/ACS).

2.7 Configuration Data Profiles

The interfaces for testing different components are specified by the EMVCo 3DS specifications[1]. Before testing can start, the Data Profiles as described in section A.1 should be configured. These profiles contain a specific ID and description, linking to predefined card ranges.

All the data profiles, except 'FrictionlessConfigurationDataNotContaining3DSMethodURL', should contain the 3DSMethodURL value for the Browser based flow.

For more information regarding the format and the values of the Configuration data profiles refer to appendix A.1

2.8 Default values

Table 2 presents the default values of fields that are used in the UL 3DS Self Test Platform.

Table 2 Default values used in the UL 3DS Self Test Platform

Field	Scenarios	Default Value	Conditional Inclusion
acquirerBIN	Default Acquirer profile for happy flow	000000999	
	Acquirer profile with a non-participating Acquirer BIN	000000000	
	Invalid format (more than 11 characters)	01234567890987	
acquirerMerchantID	Default Acquirer Merchant ID for happy flow	9876543210001	
	Acquirer Merchant ID that does not relate to the Acquirer BIN	9999999999999	
Electronic Commerce Indicator (ECI)	Happy flow value	‘00’ or ‘99’	•Required for transStatus == 'Y' and transStatus == 'A' in case the ACS is the System Under Test. •Included in the ARes message generated by the test platform for all values of transStatus in case the 3DS Server, 3DS SDK, or the DS is the System Under Test
Authentication value	Happy flow value	‘AABBCCDDEEFFAABBCCDDEEFFAAA=’ or ‘QXV0aGVudGljYXRpb24gdmFsdWU=’	Message Category = 01 and Required if transStatus = ‘Y’ or transStatus = ‘A’.

2.9 Message Extensions

As per EMVCo’s decision, testing of critical message extensions is not in scope of the platform. Any message extension received by the System Under Test with the criticalityIndicator set to ’true’ should be regarded as a non-recognized critical message extension. The test cases expect an error message to be returned in such a scenario.

Non-critical message extensions need to be handled by the Systems Under Test.

3 ACS

In Section 1.4 the four overviews were presented (App-based, Browser-based, App-based Out-of-Band and 3DS Requestor Initiated). The current chapter provides a more detailed description of the messages used during testing of an ACS as the System Under Test.

The ACS shall send/receive proprietary messages to/from the UL 3DS Test Platform containing test case relevant data not present in standard messages defined by the 3DS Specification. Information on the fields contained and required formatting can be found in Appendix A.2.

For each Flow (App-Based, Browser-Based and App-based Out-of-Band) the proprietary messages that need to be implemented are explained in detail in the following subsections. Finally, additional functionality that needs to be supported by the ACS is described.

3.1 Application Based flow

The steps described in this subsection are directly linked to the steps shown in the overview figures in Section 1.4.

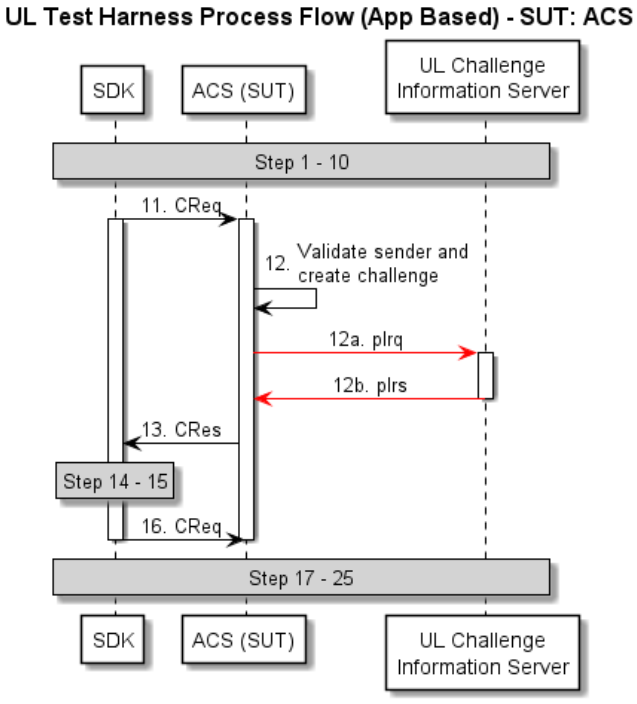


Figure 6 Detailed flow for the Application Device Channel for an ACS as System Under Test.

Steps 1-10:
Steps 1-10 take place as described in the 3-D Secure specification[1].

Steps 11-12:
After the AReq and ARes have been exchanged a CReq is sent by the SDK in Step 11. The ACS validates the sender and creates the challenge (Step 12).

Steps 12a-12b:
During step 12a, a proprietary Information Request message (pIrq) is sent by the ACS to the UL Challenge Information Server. This message:

- contains Challenge Data (correct passwords, OTP values, HTML forms, etc.) to be utilized by the UL 3DS Test Platform in successfully completed challenges sent to the Client. Details on formatting and examples can be found in appendix A.2.5.
- is sent by the ACS to the UL 3DS Test Platform during 3DS transactions utilizing Challenge Flow.

The URL where this message needs to be sent to will be available on the UL 3DS Self Test Platform or communicated directly by UL.

The UL 3DS Self Test Platform will respond with a proprietary Information Response message (pIrs) to the ACS in step 12b. This message:

- contains confirmation that the data was successfully received. Details on formatting and examples can be found in appendix A.2.6.
- is sent from UL 3DS Test Platform in response to pIrq.

The pIrq and pIrs message pair may be repeated as necessary, supplying the successful Challenge Data for each iteration of the CReq/CRes exchange. If no pIrq is sent for a subsequent interaction, the old values will be reused.

Steps 13:
A Challenge Response (CRes) is sent back (Step 13) to the SDK.

Steps 14-16:
The information received in the pIrq is used to populate the required fields and is submitted using a second CReq message according to the 3-D Secure Specification[1].

Steps 17-25:
The remaining steps are completed according to the 3-D Secure Specification[1].

3.2 Browser Based flow

The steps described in this subsection are directly linked to the steps shown in the overview figures in Section 1.4.

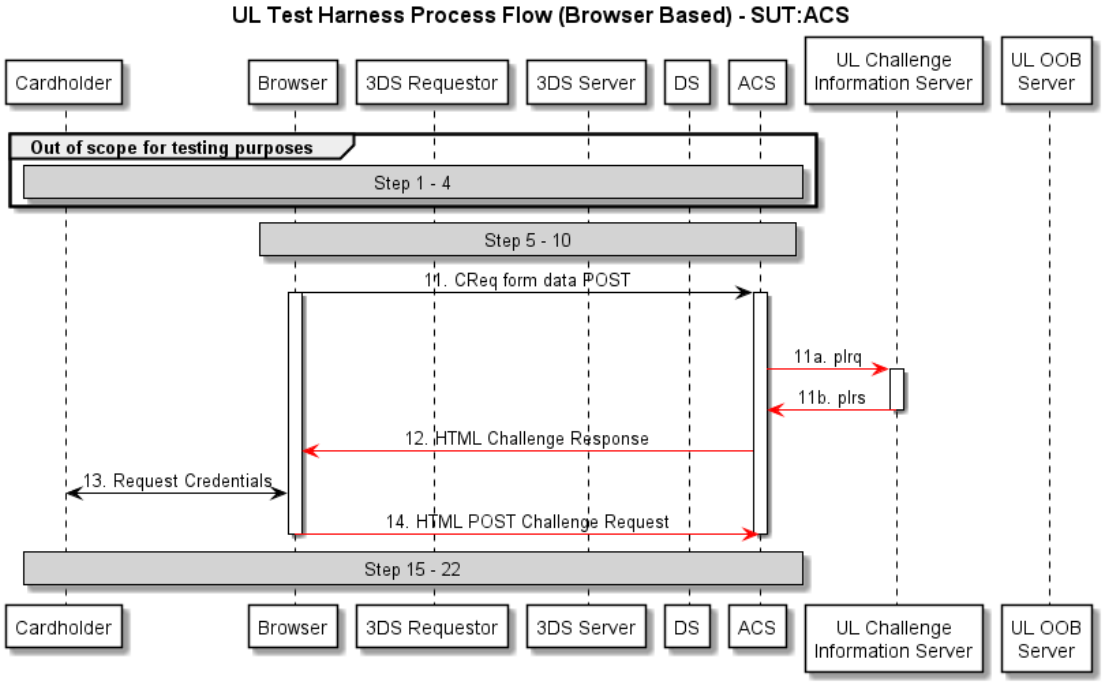


Figure 7 Detailed flow for the Browser Device Channel for an ACS as System Under Test

Steps 1-4:
Steps 1 to 4 take place as described in the 3-D Secure specification[1]. 3DS Method URL functionality is mandatory according to the EMVCo specification, but will not be considered for testing purposes in the platform.

Steps 5-10:
Steps 5-10 take place as described in the 3-D Secure specification[1].

Step 11:
A Challenge Request is sent from the Requestor Environment to the ACS.
For more details regarding the CReq HTTP POST form please refer to the example given in section A.4.1.

Steps 11a-11b:
During step 12a, a proprietary Information Request message (pIrq) is sent by the ACS to the UL Challenge Information Server. This message:

- contains Challenge Data (correct passwords, OTP values etc.) to be utilized by UL 3DS Test Platform in successfully completed challenges sent to the Client. The ACS should provide the form fields that are expected in the next CReq. Details on formatting and examples can be found in appendix A.2.5.
- is sent by ACS to UL 3DS Test Platform during 3DS transactions utilizing Challenge Flow.

The URL where this message needs to be sent to will be available on the UL 3DS Self Test Platform or communicated directly by UL.

The UL 3DS Self Test Platform will respond with a proprietary Information Response message (pIrs) to the ACS in step 12b. This message:

- contains confirmation that the data was successfully received. Details on formatting and examples can be found in appendix A.2.6
- is sent from UL 3DS Self Test Platform in response to pIrq.

The pIrq and pIrs message pair may be repeated as necessary, supplying the successful Challenge Data for each iteration of the CReq/CRes exchange. If no pIrq is sent for a subsequent interaction, the old values will be reused.

Step 12:
An HTML Challenge Response is sent from the ACS to the UL 3DS Self Test Platform in response to the CReq of step 11. This contains the challenge form that is presented to the client. The Challenge Response HTML should be formatted as described in the 3-D Secure Specification.

Steps 13-14:
The information received in the pIrq is used to populate the challenge form which is subsequently submitted to the ACS.

Steps 15-22:
The remaining steps are completed according to the 3-D Secure Specification[1].

Note: In Step 21, The ACS needs to format the final CRes message as per requirements Req 138 and Req 139 of the EMVCo Specification[1]. The ACS should send the formatted CRes message in an HTML page to the UL Platform.

For more details regarding the final CReq HTML page please refer to the example given in section A.4.2.

3.3 Application based Out-of-Band flow

The steps described in this subsection are directly linked to the steps shown in the overview figures in Section 1.4.

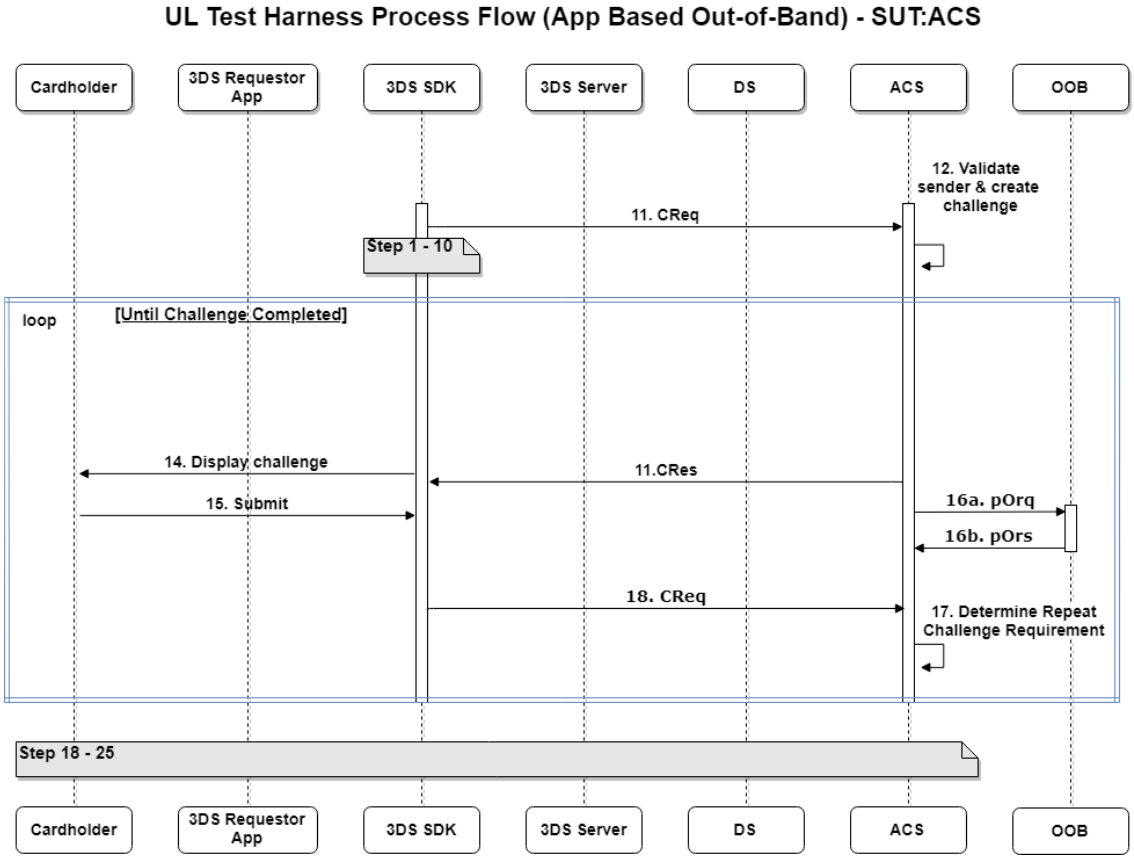


Figure 8 Detailed flow for the SDK Device Channel for an ACS as System Under Test

Steps 1-12:
Steps 1-12 take place as described in the 3-D Secure specifications[1].

Steps 13-16:
In Step 13 a Challenge Response is sent from the ACS to the Requestor Environment as described in the 3-D Secure specifications[1].

Note: For test cases referring to Application based Out-of-Band that are using HTML ACS UI Type (acsUiType=05) the SUT needs to send a pIrq with the p_formValues_APP data element to our Challenge Info Server. The aforementioned exchange of pIrq and pIrs shall happen as Step 13a and 13b respectively.

Step 16a:
After a CReq is received from the SDK (Step 16), ACS sends a proprietary Out-of-Band Request (pOrq) to the UL 3DS Self Test Platform. This message is used to simulate the Out-of-Band functionality described in the 3-D Secure Specification[1]. The pOrq:

- is sent by ACS to UL 3DS Self Test Platform during Challenge Flow in the case of Out-of-Band authentication.

- requests the UL 3DS Self Test Platform to return the result of Out-of-Band authentication.

The URL where this message needs to be sent to will be available on the UL 3DS Self Test Platform or communicated directly by UL. Details on formatting and examples can be found in appendix A.2.7

Step 16b:
In response to the pOrq, a proprietary Out-of-Band Response (pOrs) is sent from the UL 3DS Self Test Platform to the ACS. The pOrs:

- is sent from the UL 3DS Self Test Platform to ACS during Challenge Flow in the case of Out of Band Authentication.
- contains the result of out-of-band authentication. The ACS is expected to use this information in the same way as it would in a real life situation. Details on formatting and examples can be found in appendix A.2.8.

The pOrq and pOrs message pair shall be repeated for each challenge attempt.

Steps 17-25:
Steps 17-25, take place as described in the 3-D Secure specifications[1].

3.4 Decoupled authentication flow

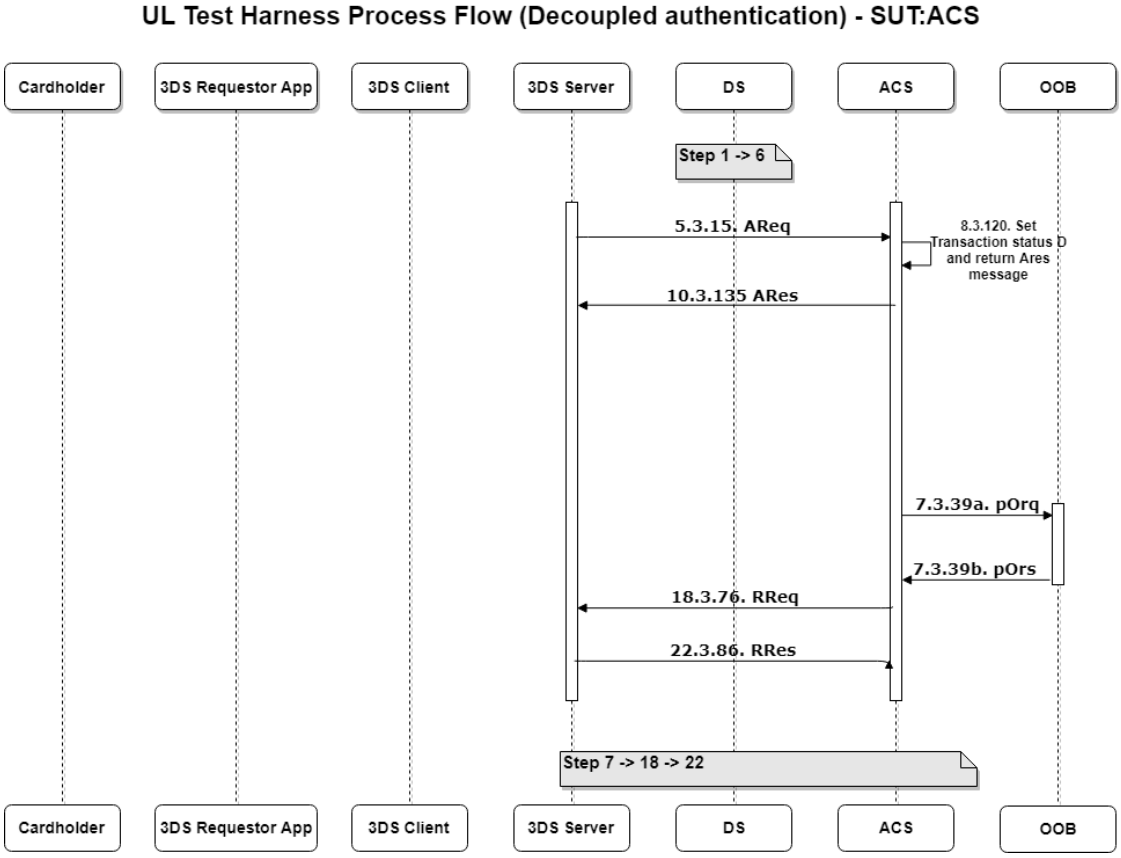


Figure 9 Detailed flow for the decoupled authentication flow for an ACS as System Under Test

Steps 1-6:
Steps 1-6 take place as described in the 3-D Secure specifications[1].

Step 7:
The ACS decides that Decoupled Authentication will be used to authenticate the cardholder and returns an ARes message with Transaction Status 'D' to the DS.

Steps 8 - 10:
In Step 8 till 10 an ARes message is sent from the ACS to the Requestor Environment as described in the 3-D Secure specifications[1].

After the ACS has returned the ARes message to the DS it reaches out to the OOB Server using the proprietary Out-of-Band Request (pOrq) message to obtain the result of the cardholder authentication using the Decoupled Authentication channel (refer to Requirement 322b in the 3-D Secure specifications[1]).

Step 18:
Steps 18, takes place as described in the 3-D Secure specifications[1].

An RReq message is sent immediately upon obtaining an authentication result (whether successful or not). Note, that for testing purposes it is not feasible to use a grace period of 1 hour as mentioned in the 3-D Secure specifications[1]. Therefore, the ACS has to send the RReq message within the 3DS Requestor Decoupled Max Time provided in the AReq message.

Steps 19-22:
Steps 19 till 22, take place as described in the 3-D Secure specifications[1].

Note: 3-D Secure processing completes for Decoupled Authentication transactions in this step.
Note: for the purpose of this section the 01-APP Device Channel flow has been used to describe the Decoupled Authentication flow. The same changes to the flow outlined in the 3-D Secure specifications[1] as described above apply to the 02-BRW and 03-3RI Device Channels.

3.5 UL Simulator Endpoints

In addition to the standard 3DS flow connections, the ACS must connect to two UL Servers – the Out of Band Simulator Server and the UL Challenge Information Server. The endpoints for these connections are available on the UL 3DS Self Test Platform, and will follow the format presented in the Table 3.

Table 3 UL Servers' Connection Endpoints.

UL Server Name	Connection Endpoints
UL Challenge Information Server	https://simulator-mutual-3ds.selftestplatform.com/{version}/info/{projectid}/
UL Out-of-Band (OOB) Server	https://simulator-mutual-3ds.selftestplatform.com/{version}/oob/{projectid}/

These endpoints follow the formatting of the standard 3DS flow endpoint URLs visible in the UL 3DS Self Test Platform on the "Configuration" screen under the "Endpoints" tab.

{version} represents the current version of the 3-D Secure specifications[1] in the format "v2.X.X" and match the version used by the Product Provider at the time of certification.

{projectid} represents the UL 3DS Self Test Platform project ID associated with your project by the UL 3DS Self Test Platform.

To connect to these endpoints, the use of a client certificate is required. Please check Section 2.5 and Section 6.1.3.2 of the 3DS Specification[1] for details on the certificates to use.

3.6 Configuration Data Profiles

The interfaces for testing the DS are specified by the EMVCo 3DS specifications[1]. Before testing can start, the Data Profiles as described in section A.1 should be configured. These profiles contain a specific ID and description, linking to predefined card ranges.

For more information regarding the format and the values of the Configuration data profiles refer to appendix A.1.

3.7 Additional Functionalities

In addition to recognizing and responding to UL Proprietary message types, the following additional functionalities must be supported by the ACS:

- The ACS shall behave in a pre-determined manner based on a supplied response (CReq or HTML) in a Challenge Flow. This includes resending of a challenge if required.
- UL shall be supplied with the POST data that needs to be sent as challenge response via the pIrq message.
- Provide the ACS public key to be signed by the UL DS CA.
- ACS certificates shall be signed by the UL DS CA using the web platform.
- Use a maximum of three (3) transactions for the interaction counter.
- For the 3RI and NPA channels the Authentication Value will not be checked

4 DS

As the interfaces for the DS are specified by the 3-D Secure specification there is minor additional configuration needed for the DS. Transaction Status Values and Configuration Data profiles must be implemented as described below.

There are no proprietary messages required to be implemented for a DS as the SUT.

4.1 Application, Browser, Out-of-Band based flow

The steps described in this subsection are directly linked to the steps shown in the overview figures in Section 1.4. Figure presents the App Based flow when the system under Test is the DS. As can be concluded from the diagram, there are no proprietary messages required to be implemented when a DS is the SUT.

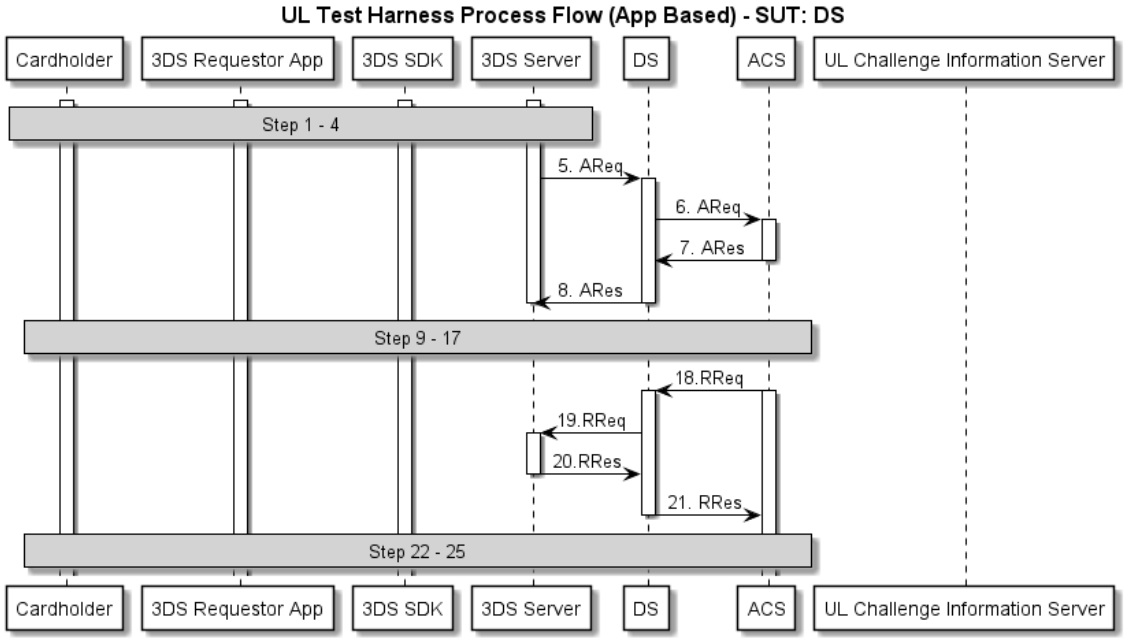


Figure 10 Detailed flow for the Application Device Channel for a DS as System Under Test.

On the same basis, the browser based flow, the application based Out-of-Band and the 3DS Requestor Initiated do not require any implementation of proprietary messages when a DS is to be connected to the UL 3DS Self Test Platform.

4.2 Configuration Data Profiles

The interfaces for testing the DS are specified by the EMVCo 3DS specifications[1]. Before testing can start, the Data Profiles as described in section A.1 should be configured. These profiles contain a specific ID and description, linking to predefined card ranges.

All the data profiles, except 'FrictionlessConfigurationDataNotContaining3DSMethodURL', should contain the 3DSMethodURL value for the Browser based flow.

For more information regarding the format and the values of the Configuration data profiles refer to appendix A.1

4.3 Operator ID Values

In addition to the configuration data profiles, the DS must also recognize two operator ID values which will be utilized by certain test cases. The fields threeDSServerOperatorID and acsOperatorID are DS assigned values that may be assigned to a 3DS Server and ACS, respectively – as per the EMVCo Specifications[1].

The values to be used for the Operators ID are presented in Table 4.

Table 4 Operator ID Values

Data Element / Field Name	Value
threeDSServerOperatorID	threeDSServerOperatorUL
acsOperatorID	acsOperatorUL

4.4 Merchant Category Code

To test the Error Code 306, the test cases expect a DS to reject merchant category code (mcc) value equal to 0000.

5 SDK

To connect to the UL 3DS Self Test Platform, the SDK needs to implement the following functionality as a test harness.

5.1 UL Reference Application Integration

The SDK must integrate with the UL Reference Application to facilitate automation of test case execution for the supported operating systems (initially Android and iOS). This Reference Application will be communicated with the Product Provider directly during the set-up phase.

Besides the integration with the Reference Application the SDK needs to implement Challenge Listeners and allow certain security permissions to facilitate automation during testing.

Figure presents the exchanged messaged between the 3DS SDK, the UL Reference App and the rest of the ecosystem when the 3DS SDK is the SUT. Moreover, when the UL Reference application is used, there is no need of further implementation for the proprietary messages, as they are being handled from the UL Reference application.

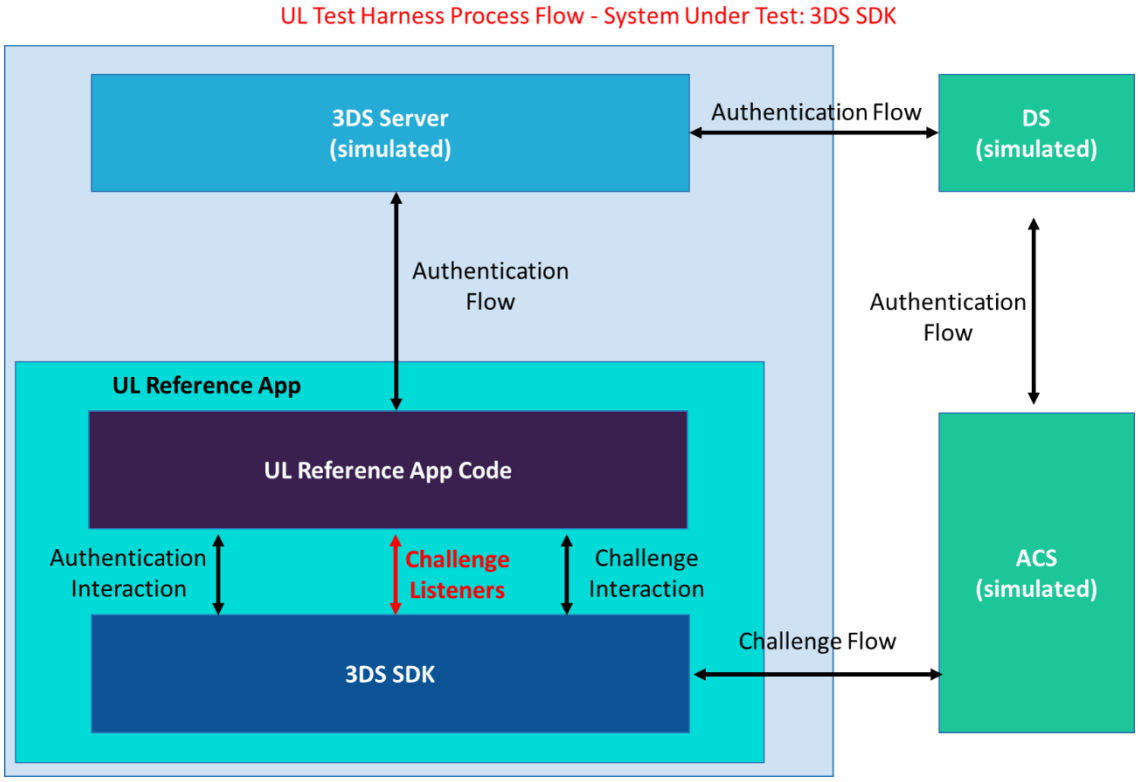


Figure 11 Flow diagram when a 3DS SDK is the System Under Test.

5.1.1 Supported versions

The UL Reference Application is available and supported for the following development platforms and versions/levels.

iOS:

- Please refer to the iOS 3DS Reference App Developers Guide for the list of supported Swift versions.

Android:

- Please refer to the Android 3DS Reference App Developers Guide for the Minimum API Level required.

5.1.2 Challenge Listeners

The purpose of the Challenge Listeners is to extract the required information supplied during challenges and allow the UL 3DS Self Test Platform to properly populate/validate challenge data fields as required during testing. As each challenge type functions differently, there is a unique Listener per challenge type, as detailed in the iOS and Android 3DS Reference App Developers Guides.

Chapter 7 in the 3DS Reference App Developers Guides for both iOS and Android also contains code samples for the various Challenge Listeners.

5.1.3 Security permissions

Certain security permissions are required to allow for specific functionality needed during testing. The SDK shall grant the following permissions to the UL Reference Application during testing:

- To capture screenshots. This means security requirement SE08 as described in the 3DS SDK Specification[3] will be intentionally bypassed. Please also see section 3.5.10 of the 3DS Technical Guide[2] for more information on disabling screen capture. SE08 should only be bypassed during EMVCo functional testing.
- To inject data into challenge text views using the Challenge Listeners. Please see the Challenge Listener example codes in the 3DS Reference App Developers Guide for iOS and Android for more information.

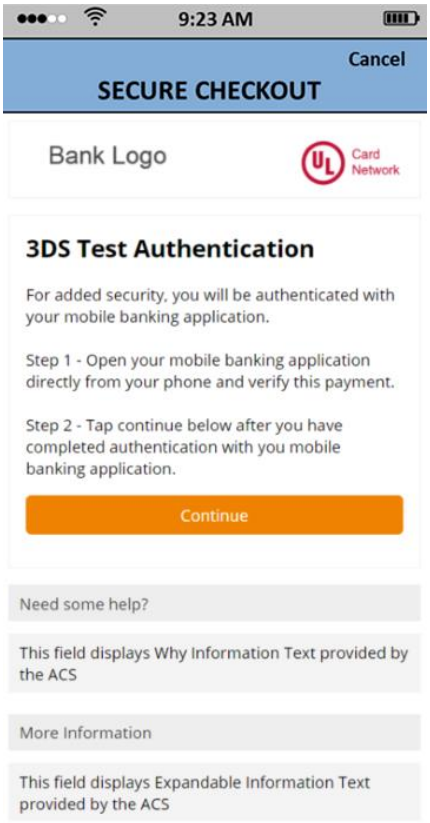
5.2 Additional Functionalities

In addition to utilizing the Challenge Listeners described in Section 5.1.2, the SDK shall:

- Be capable of retrieving/loading the UL DS CA public key. Values for the RSA and EC Public keys are listed in A.5
- In case that the SDK is based on iOS or Android OS solution, the Product Provider should use the provided UL Reference Application. In any other case of non-iOS or non-Android solutions the Product Provider should create their own reference application.

5.3 Visual Validations

To validate that the 3DS SDK renders the proper UI based on the CRes message received from the ACS, the Test Plan contains a number of UI related test cases that require visual validation. To avoid that the screenshots captured by the UL Reference App do not include all UI elements due to the screen size of the (mobile) device used for testing, it is recommended that the Product Provider first executes test case TC_SDK_10026_001 to ensure the captured screenshot contains all UI data elements provided by the ACS as shown in the figure below:



In case not all UI elements (including the expandable text areas at the bottom of the screen) fit on the screen, it is strongly recommended to use a device with a larger screen size and/or higher resolution to avoid delays during the review of the test results performed by the Test Lab.

Furthermore, it is advised to pay attention during the pre-compliance testing, that all screenshots present in the Test Platform – either automatically captured by the UL Reference App, or manually uploaded by the Product Provider – always contain all UI elements.

6 3DS Server

In Section 1.4 the four overviews were presented (App-based, Browser-based, Out-of-Band and 3DS Requestor Initiated). The current chapter provides a detailed description of the messages used during testing when the 3DS Server is considered the System Under Test.

The 3DS Server shall send/receive proprietary messages to/from the UL 3DS Self Test Platform containing test case relevant data not present in standard messages defined by the 3DS Specification. Information on the fields contained and required formatting can be found in Appendix A.2.

For each Flow (App-based, Browser-based and Out-of-Band) the proprietary messages that need to be implemented are explained in detail in the following subsections. Finally, additional functionality that needs to be supported by the 3DS Server is described.

6.1 Application based flow

The steps described in this subsection are directly linked to the steps shown in the overview of Figure in Section 1.4.

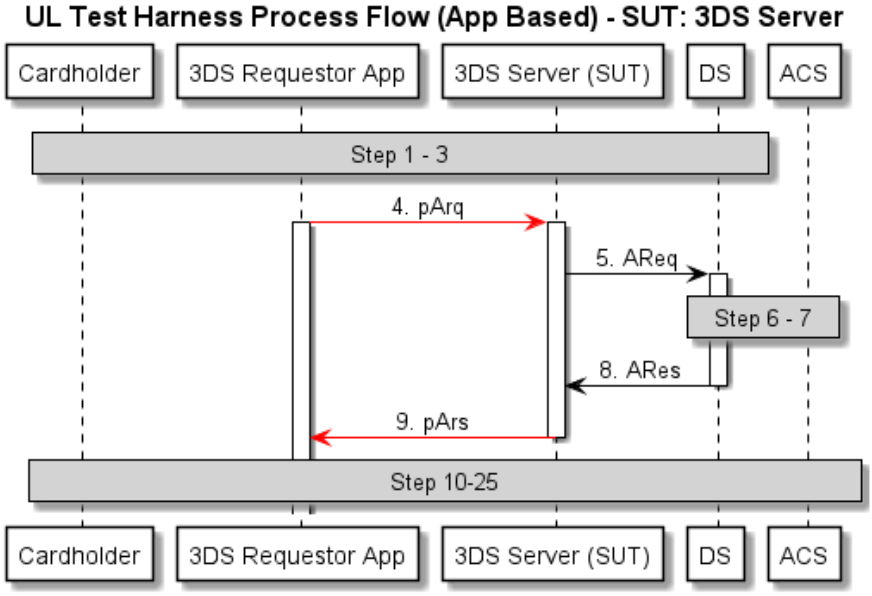


Figure 12 Detailed flow for the Application Device Channel for 3DS Server as System Under Test

Steps 1-3:
Step 1-3 take place as described in the 3-D Secure specification.

Step 4:
A proprietary Authentication Request (pArq) is sent from the UL 3DS Self Test Platform to the 3DS Server. The pArq:

- contains the 3DS Authentication data required by 3DS Server to build AReq message. Details on formatting and examples can be found in appendix A.2.
- is sent from UL 3DS Self Test Platform to 3DS Server prior to 3DS Server constructing AReq.

Steps 5-8:
Step 5-8 take place as described in the 3-D Secure specification[1].

For testing purposes, as per Requirement 40 in the 3-D Secure specification[1], if the transaction status in the received ARes = C, the 3DS Server shall collect the necessary information for the challenge from the ARes message to be sent down to the 3DS Requestor Environment in the form of the proprietary Authentication Response message (Step 9). Note that there are some exceptions to this rule in case of certain test cases such as where the 3DS Requestor Challenge Indicator is set to 02 (No Challenge Requested) and the Transaction status = 'C', or tests where the transaction status is not 'C". In these exceptional cases, the pArs sent to the 3DS Requestor environment must not contain data elements that are required only when the Transaction Status = C such as the acsSignedContent or acsURL.

Step 9:
A proprietary Authentication Response (pArs) is sent from the 3DS Server to the UL 3DS Self Test Platform. The pArs:

- contains fields required by the UL 3DS Self Test Platform to continue the transaction.
- is sent by 3DS Server to the UL 3DS Self Test Platform after 3DS Server receives ARes message from the DS.

Steps 10-25:
Steps 10-25 take place as described in the 3-D Secure specification[1].

6.2 Browser based flow

The steps described in this subsection are directly linked to the steps shown in the overview figure in Section 1.4.

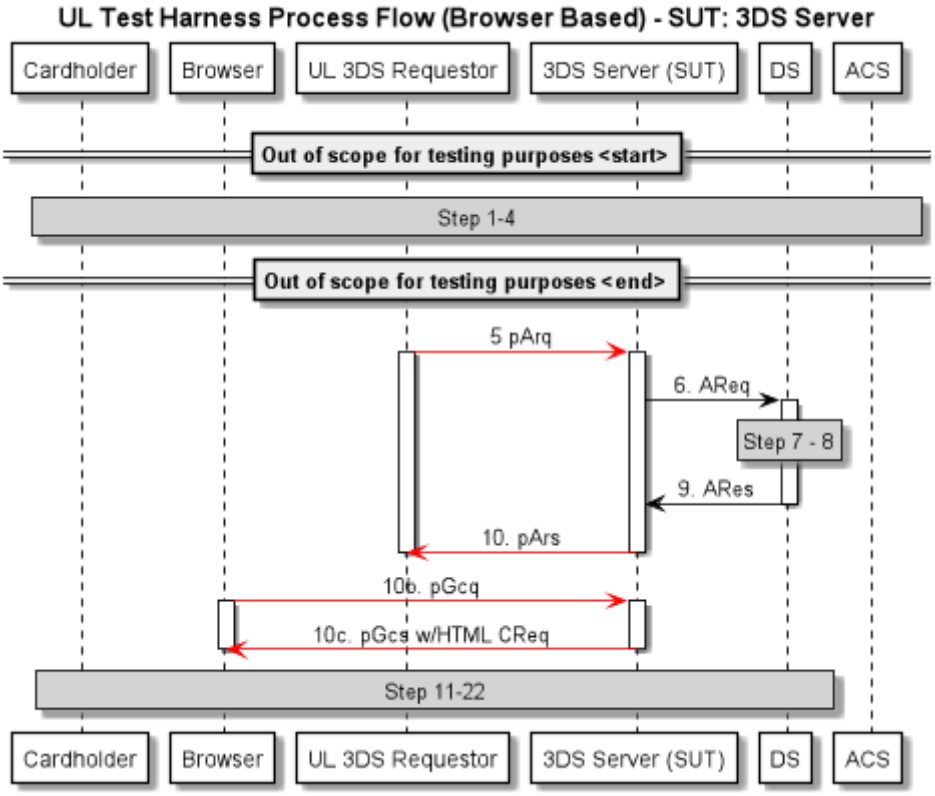


Figure 13 Detailed flow for the Browser Device Channel for a 3DS Server as System Under Test.

Steps 1-4:
Steps 1 to 4 take place as described in the 3-D Secure specification[1]. 3DS Method URL functionality is mandatory according to the EMVCo specification, but will not be considered for testing purposes in the platform. The 3DS Server system under test should use the value of 'U' for the 3DS Method Completion Indicator which indicates that the 3DS Method URL does not exist.

Step 5:
An Authentication Requests (pArq) is sent from the UL 3DS Self Test Platform to the 3DS Server. The pArq:

- contains the 3DS Authentication data required by 3DS Server to build AReq message. Details on formatting and examples can be found in appendix A.2.
- is sent from UL 3DS Self Test Platform to 3DS Server prior to 3DS Server constructing AReq.

Steps 6-9:
Step 6-9 take place as described in the 3-D Secure specification[1].

Step 10:
A proprietary Authentication Response (pArs) is sent from the 3DS Server to the UL 3DS Self Test Platform. The pArs:

- contains fields required by the UL 3DS Self Test Platform to continue the transaction.
- is sent by 3DS Server to the UL 3DS Self Test Platform after 3DS Server receives ARes message from the DS.

Steps 10b-10c:
To simulate the behavior described in Requirement 117b-e[1], a proprietary Get Challenge request (pGcq) is sent from the UL 3DS Self Test Platform to the 3DS Server. The 3DS Server is expected to respond with a Get Challenge response, containing the initial CReq that needs to be forwarded to the ACS (in normal operation via the cardholder browser). Details on formatting and examples can be found in appendix A.2.3 and A.2.4.

Steps 12-22:
According to the EMV® 3-D Secure specifications[1].

6.3 Application based Out-of-Band flow

For the 3DS Server as a System Under Test, the Out-of-Band flow is the same as non-Out-of-Band flow and is therefore not repeated. Please see Section 6.1 for more details.

6.4 3DS Requestor Initiated (3RI) flow

The steps described in this subsection are directly linked to the steps shown in the overview figures in Section 1.4.

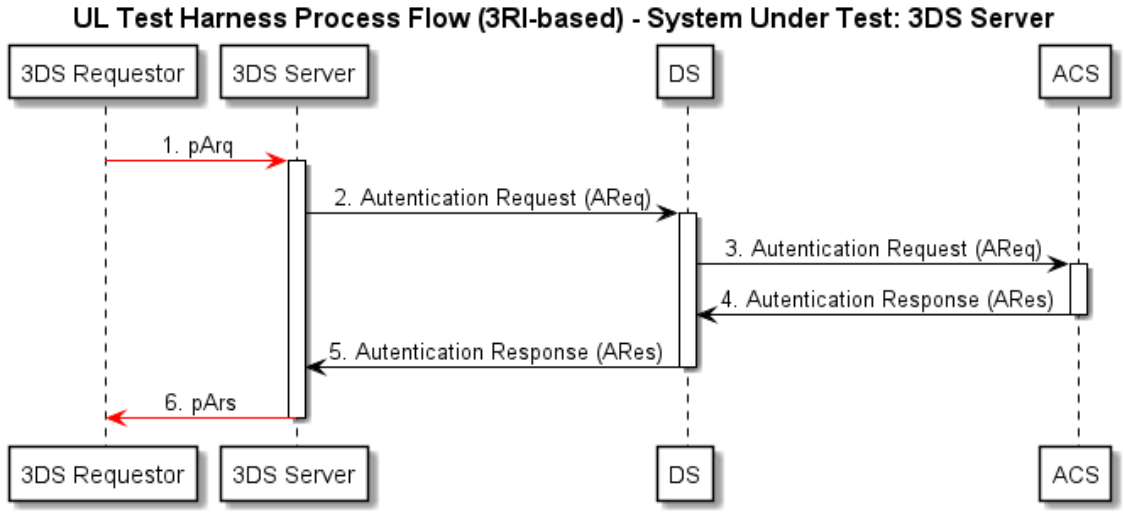


Figure 14 Detailed flow for the 3RI flow for a 3DS Server as System Under Test.

Step 1:
A proprietary Authentication Request (pArq) is sent from the UL 3DS Self Test Platform to the 3DS Server. The pArq:

- contains the 3DS Authentication data required by 3DS Server to build AReq message. Details on formatting and examples can be found in appendix A.2.

- is sent from UL 3DS Self Test Platform to 3DS Server prior to 3DS Server constructing AReq.

Steps 2-5:
Steps 2-5 take place as described in the 3-D Secure specification[1].

Step 6:
A proprietary Authentication Response (pArs) is sent from the 3DS Server to the UL 3DS Self Test Platform. The pArs:

- contains fields required by the UL 3DS Self Test Platform to continue the transaction.
- is sent by 3DS Server to the UL 3DS Self Test Platform after 3DS Server receives ARes message from the DS.

6.5 PReq/PRes flow

The steps described in this subsection define the proprietary messages during the PReq/PRes exchange between the 3DS Server (as SUT) and the DS. The flow of these messages is represented in Figure **.**

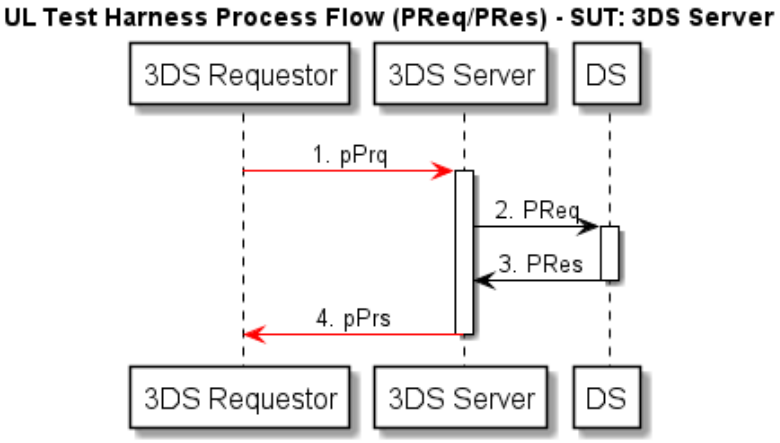


Figure 15 UL TS Test Harness Process Flow (PReq/PRes) System Under Test: 3DS Server

The pPrq and corresponding pPrs will only be used when required explicitly by a test case. This is on top of normal PReq/PRes message exchange defined by the EMVCo specification.

Step 1:
A proprietary Preparation Request (pPrq) is sent from the UL 3DS Self Test Platform to the 3DS Server. The pPrq:

- prompts 3DS Server to generate PReq message for each known DS on command, including outside the normal 3DS authentication flow if required.
- is sent from the UL 3DS Self Test Platform to 3DS Server.

Steps 2-3:
Steps 2-3 take place as described in the 3-D Secure specification[1].

Step 4:
A proprietary Preparation Response (pPrs) is sent to the UL 3DS Self Test Platform from the 3DS Server. The pPrs:

- contains confirmation that a PRes message was received by the 3DS Server.

- is sent by 3DS Server to UL 3DS Self Test Platform in response to pPrq.

6.5.1 Serial Number processing

Although the EMVCo® 3-D Secure Protocol and Core Functions specification defines that the 3DS Server should include the last known Serial Number (as received from the DS in the PRes message) in the next PReq message sent to the DS, for testing purposes, the very first PReq message of each test case should not contain a Serial Number in order to start each test case with a clean state.

The UL 3DS Self Test Platform uses internally the so called Test Case Run ID to identify a specific test case execution which is included as 'x-ul-testcaserun-id' in the HTTP header. As it is the same identifier for each message sent as part of the same test case, a 3DS Server implementation can use it to identify whether a Serial Number received in a PRes message belongs to the same test case and hence is still valid when generating a subsequent PReq message as in that case the 'x-ul-testcaserun-id' sent together with the pPrq message will be the same as the one received together with the PRes message.

6.5.2 Test Case Identification

Whenever the DS within the UL 3DS Self Test Platform receives a request message from the 3DS Server (acting as System Under Test), the UL Test Platform tries to link the received message to a test case. For this purpose, various unique data elements included in the received message are used (e.g. the SDK Transaction ID or the 3DS Requestor URL). However, as the number of data elements included in a PReq message is limited, the options to uniquely identify an incoming PReq message are limited as well.
This Test Harness specification therefore provides two possible means to circumvent the beforementioned problem by adding both the threeDSServerTransID and the threeDSRequestorUL data elements to the pPrq message defined in Annex A.2.9. In order to allow the UL Test Platform to correctly identify the test case to which the received PReq message belongs, the 3DS Server has to forward at least one of these two data elements in the PReq message sent to the DS.

6.6 Error messages between UL 3DS Requestor and 3DS Server

Error messages shall be sent to the UL 3DS Requestor by the 3DS Server for all scenarios described in the EMV 3-D Secure Core specification[1]. Including, but not limited to, section 5.9.4, 5.9.9 and requirements 229 and 271 of the specification.

Error messages sent from the 3DS Server to the UL 3DS Requestor will be handled as follows:

> As a response to the pArq or pPrq message over the connection that was set up for the pArq/pAre or pPrq/pPre message exchange (instead of the pAre/pPre message). For example, if an Ares or PRes is received from the DS that contains an error (incorrect value etc.) or if an Erro message is received from the DS.

The error messages shall be formatted according to sections A.5.5, B.10 and table A.4 of the EMV 3-D Secure Core specification[1].

6.7 Additional Functionalities

In addition to recognizing and responding to UL Proprietary message types and error messages, the following additional functionalities must be supported by the 3DS Server:

- Use information from the 3DS Requestor (found in the pArq messages) to build AReq message.

References

Ref.	Title	Author	Version	Date
[1]	EMV® 3-D Secure Protocol and Core Functions Specification	EMVCo, LLC	2.1.0 2.2.0	10-30-2017 December 2018
[2]	EMV® 3-D Secure – SDK Technical Guide	EMVCo, LLC	2.1.0	10-31-2017
[3]	EMV® 3-D Secure – SDK Specification	EMVCo, LLC	2.1.0 2.2.0	10-31-2017 December 2018
[4]	3DS Self Test Platform – iOS 3DS Reference App Developers Guide	UL	1.4	March 2019
[5]	3DS Self Test Platform – Android 3DS Reference App Developers Guide	UL	1.2	April 2019

A.1 Configuration Data Profiles

A.1.1 Format

More specifically, regarding the Account Range number, the following syntax was used:
Account Range: **NNNNNNCCXXXXXXXXXY,** in which:

- **N** – BIN (6 digits)
- **C** – number of the configuration data profiles (2 digits)
- **X** – identify the account (4 to 9 digits)
- **Y** – Check Digit based on the mod10 (Luhn) algorithm.

Each card range will use account numbers with a fixed length ranging between 13 and 19 digits, as is allowed by the EMV 3-D Secure specifications[1].

Note: The account numbers used by the test platform will **not** pass the mod10 algorithm (also known as the Luhn Algorithm) to avoid the usage of production grade account numbers for testing purposes.

A.1.2 Ranges

Table 5 Card Ranges

№	ID	Account range		Description
		Start	End	
1.	UnsupportedDeviceConfigurationData	6543200100000	6543200199999	Process as if the device is unsupported
2.	AuthenticationUnsupportedConfigurationData	65432102000000	65432102999999	Process as if the authentication is not available due to the Configuration Data not representing a valid 3DS Account
3.	<removed>	<removed>	<removed>	<removed>
4.	FrictionlessConfigurationData	7654310400000000	7654310499999999	Perform Frictionless authentication flow, transStatus = Y
5.	CardholderNotAuth	7654320500000000	7654320599999999	Process such that transaction is not authenticated, transStatus = N
6.	CardholderCouldNotAuth	7654340600000000	7654340699999999	Process as if authentication could not be performed, transStatus = U
7.	CardholderAttemptedAuth	7654350700000000	7654350799999999	Process as if authentication could not be performed, transStatus = A
8.	CardholderRejected	7654360800000000	7654360899999999	Process as if authentication was available and rejected, transStatus = R
9.	CardholderChallengeAcsRenderingType0101	7654370900000000	7654370999999999	Perform Challenge flow using acsRenderingType ["01", "01"]

10.	CardholderChallengeAcsRenderingType0102	7654381000000000	7654381099999999	Perform Challenge flow using acsRenderingType ["01", "02"]
11.	CardholderChallengeAcsRenderingType0103	7654391100000000	7654391199999999	Perform Challenge flow using acsRenderingType ["01", "03"]
12.	CardholderChallengeAcsRenderingType0104	8765411200000000	8765411299999999	Perform Challenge flow using acsRenderingType ["01", "04"]
13.	<removed>	<removed>	<removed>	<removed>
14.	CardholderChallengeAcsRenderingType0201	8765431400000000	8765431499999999	Perform Challenge flow using acsRenderingType ["02", "01"]
15.	CardholderChallengeAcsRenderingType0202	8765441500000000	8765441599999999	Perform Challenge flow using acsRenderingType ["02", "02"]
16.	CardholderChallengeAcsRenderingType0203	8765451600000000	8765451699999999	Perform Challenge flow using acsRenderingType ["02", "03"]
17.	CardholderChallengeAcsRenderingType0204	8765461700000000	8765461799999999	Perform Challenge flow using acsRenderingType ["02", "04"]
18.	CardholderChallengeAcsRenderingType0205	8765471800000000	8765471899999999	Perform Challenge flow using acsRenderingType ["02", "05"]
19.	CardholderOutOfRange	8765481900000000	8765481999999999	Process as if Cardholder Account Number is not in a participating account range, transStatus = U
20.	CardholderOutOfRangeACS	8765492000000000	8765492099999999	DS should process as if Cardholder Account Number is not in an account range that has an ACS capable of processing 3-D Secure messages, transStatus = U
21.	<removed>	<removed>	<removed>	<removed>
22.	<removed>	<removed>	<removed>	<removed>
23.	<removed>	<removed>	<removed>	<removed>
24.	CardholderAuthNotAvailable	1876542400000000000	1876542499999999999	Process as if the Authentication is not available

				or cannot be completed for the cardholder.
25.	CardholderFrictionlessNotContaining3DSMethodURL	8765422500000000	8765422599999999	Perform Frictionless authentication flow, transStatus = Y, but the 3DS Method URL is not available in the PRes (only used by DS simulator in the UL 3DS Self Test Platform to construct the PRes message).
26.	CardholderFrictionlessNotContainingDSProtocolVersion	9876512600000000	9876512699999999	Perform Frictionless authentication flow, transStatus = Y, but the dsEndProtocolVerison and dsStartProtocolVersion are not available in the entry for this account range in the Card Range Data returned in the PRes message (only used by DS simulator in the UL 3DS Self Test Platform to construct the PRes message).
27.	CardholderChallenge	765430270000000	765439279999999	Perform Challenge flow. In the 01-APP device channel, the choice of acsRenderingType is up to the discretion of the System Under Test within the following restrictions: acsRenderingType indicates that a Native UI is being used, i.e. “acsInterface” equals to “01”.
28.	CardholderChallengeHTML	9876522800000000	9876522899999999	Perform Challenge flow. In the 01-APP device channel, the

				choice of acsRenderingType is up to the discretion of the System Under Test within the following restrictions: acsRenderingType indicates that an HTML UI is being used, i.e. “acsInterface” equals to “02”. ACS Information Indicator is set to [“01”, “02”, “04”] in the Card Range Data (if 3-D Secure Protocol Version Number 2.2.0 is used).
29.	CardholderDecoupledAuthentication	9876532900000000	9876532999999999	Perform decoupled authentication, transStatus = D (in ARes message) (**only for 3-D Secure Protocol Version Number 2.2.0**). ACS Information Indicator is set to [“01”, “02”, “03”, “04”] in the Card Range Data.
30.	CardholderForInformation	9876543000000000	9876543099999999	Process for informational purpose, transStatus = I (in ARes message) (**only for 3-D Secure Protocol Version Number 2.2.0**).
31.	CardholderDecoupledMaxTimeExpired	9876553100000000	9876553199999999	Perform Decoupled Authentication (transStatus = D (in ARes message) in which the cardholder does not authenticate within the 3DS Requestor Decoupled Max Time. The ACS sends

				the RReq message within the 30 second grace period maintained by the 3DS Server (**only for 3-D Secure Protocol Version Number 2.2.0**). ACS Information Indicator is set to [“01”, “02”, “03”, “04”] in the Card Range Data.
32.	CardholderDecoupledAuthAndWhiteListingNotSupported	9876563200000000	9876563299999999	The ACS does not support Decoupled Authentication and Whitelisting for this account range (**only for 3-D Secure Protocol Version Number 2.2.0**). Return transStatus = C if challenge is required. ACS Information Indicator is set to [“01”, “02”] in the Card Range Data.

Notes:
In case the card ranges are included in the ‘cardRangeData’ data element in the PRes message:

1. All the Card Ranges should include threeDSMethodURL except from CardholderFrictionlessNotContaining3DSMethodURL (No.25) in which the threeDSMethodURL should be absent
2. All the Card Ranges should include the relevant Protocol Version elements except from CardholderFrictionlessNotContainingDSProtocolVersion (No.26), in which the dsEndProtocolVerison and dsStartProtocolVersion should be absent. The values should be in line with the version of EMV® 3-D Secure Protocol and Core Functions specification supported by the System Under Test.
3. As account numbers from card ranges number 19 and 20 are to be treated as ‘not participating’, a DS as System Under Test most likely does not want to list these two ranges in the ‘cardRangeData’ data element in the PRes messages. Absence or presence of these two ranges will not affect the outcome of the DS specific test cases available in the platform.

4. Unless otherwise mentioned in the table above, the optional ACS Information Indicator shall be either absent or at least set to [“01”, “02”, “04”] in the relevant entries of the ‘cardRangeData’ data element.

Furthermore, receiving systems should consider any account numbers that are not part of the card ranges listed in this section as not belonging to the Issuer (hence resulting in Error Code ‘305’).

A.2 Proprietary messages

This section contains a detailed list of all proprietary messages used by the UL 3DS Self Test Platform for communication during testing. For each of the messages an overview of the data elements is shown as well as an example of the message.

A.2.1 Proprietary Authentication Request (pArq)

№	Field	Presence	Type/Format	Accepted Value
1	acctNumber	R	According to EMVCo 3DS Spec	According to EMVCo 3DS Spec
2	cardExpiryDate	R	According to EMVCo 3DS Spec	According to EMVCo 3DS Spec
3	deviceChannel	R	According to EMVCo 3DS Spec	According to EMVCo 3DS Spec
4	messageCategory	R	According to EMVCo 3DS Spec	According to EMVCo 3DS Spec
5	messageType	R	According to EMVCo 3DS Spec	pArq
6	messageVersion	R	According to EMVCo 3DS Spec	According to EMVCo 3DS Spec
7	p_messageVersion	R	1.0.5	Starting value: 1.0.0
8	threeDSRequestorID	R	According to EMVCo 3DS Spec	According to EMVCo 3DS Spec
9	threeDSRequestorName	R	According to EMVCo 3DS Spec	According to EMVCo 3DS Spec
10	threeDSRequestorURL	R	According to EMVCo 3DS Spec	According to EMVCo 3DS Spec
11	acquirerBIN	C	According to EMVCo 3DS Spec	According to EMVCo 3DS Spec
12	acquirerMerchantID	C	According to EMVCo 3DS Spec	According to EMVCo 3DS Spec
13	addrMatch	C	According to EMVCo 3DS Spec	According to EMVCo 3DS Spec
14	billAddrCity	C	According to EMVCo 3DS Spec	According to EMVCo 3DS Spec
15	billAddrCountry	C	According to EMVCo 3DS Spec	According to EMVCo 3DS Spec
16	billAddrLine1	C	According to EMVCo 3DS Spec	According to EMVCo 3DS Spec
17	billAddrLine2	C	According to EMVCo 3DS Spec	According to EMVCo 3DS Spec
18	billAddrLine3	C	According to EMVCo 3DS Spec	According to EMVCo 3DS Spec
19	billAddrPostCode	C	According to EMVCo 3DS Spec	According to EMVCo 3DS Spec
20	billAddrState	C	According to EMVCo 3DS Spec	According to EMVCo 3DS Spec
21	browserAcceptHeader	C	According to EMVCo 3DS Spec	According to EMVCo 3DS Spec
22	browserColorDepth	C	According to EMVCo 3DS Spec	According to EMVCo 3DS Spec
23	browserIP	C	According to EMVCo 3DS Spec	According to EMVCo 3DS Spec
24	browserJavaEnabled	C	According to EMVCo 3DS Spec	According to EMVCo 3DS Spec
25	browserLanguage	C	According to EMVCo 3DS Spec	According to EMVCo 3DS Spec
26	browserScreenHeight	C	According to EMVCo 3DS Spec	According to EMVCo 3DS Spec
27	browserScreenWidth	C	According to EMVCo 3DS Spec	According to EMVCo 3DS Spec
28	browserTZ	C	According to EMVCo 3DS Spec	According to EMVCo 3DS Spec
29	browserUserAgent	C	According to EMVCo 3DS Spec	According to EMVCo 3DS Spec
30	cardholderName	C	According to EMVCo 3DS Spec	According to EMVCo 3DS Spec
31	deviceRenderOptions	C	According to EMVCo 3DS Spec	According to EMVCo 3DS Spec
32	email	C	According to EMVCo 3DS Spec	According to EMVCo 3DS Spec
33	homePhone	C	According to EMVCo 3DS Spec	According to EMVCo 3DS Spec
34	mcc	C	According to EMVCo 3DS Spec	According to EMVCo 3DS Spec
35	merchantCountryCode	C	According to EMVCo 3DS Spec	According to EMVCo 3DS Spec

№	Field	Presence	Type/Format	Accepted Value
36	merchantName	C	According to EMVCo 3DS Spec	According to EMVCo 3DS Spec
37	mobilePhone	C	According to EMVCo 3DS Spec	According to EMVCo 3DS Spec
38	purchaseAmount	C	According to EMVCo 3DS Spec	According to EMVCo 3DS Spec
39	purchaseCurrency	C	According to EMVCo 3DS Spec	According to EMVCo 3DS Spec
40	purchaseDate	C	According to EMVCo 3DS Spec	According to EMVCo 3DS Spec
41	purchaseExponent	C	According to EMVCo 3DS Spec	According to EMVCo 3DS Spec
42	recurringExpiry	C	According to EMVCo 3DS Spec	According to EMVCo 3DS Spec
43	recurringFrequency	C	According to EMVCo 3DS Spec	According to EMVCo 3DS Spec
44	sdkAppID	C	According to EMVCo 3DS Spec	According to EMVCo 3DS Spec
45	sdkEncData	R	According to EMVCo 3DS Spec	According to EMVCo 3DS Spec
46	sdkEphemPubKey	C	According to EMVCo 3DS Spec	According to EMVCo 3DS Spec
47	sdkReferenceNumber	C	According to EMVCo 3DS Spec	According to EMVCo 3DS Spec
48	sdkTransID	C	According to EMVCo 3DS Spec	According to EMVCo 3DS Spec
49	shipAddrCity	C	According to EMVCo 3DS Spec	According to EMVCo 3DS Spec
50	shipAddrCountry	C	According to EMVCo 3DS Spec	According to EMVCo 3DS Spec
51	shipAddrLine1	C	According to EMVCo 3DS Spec	According to EMVCo 3DS Spec
52	shipAddrLine2	C	According to EMVCo 3DS Spec	According to EMVCo 3DS Spec
53	shipAddrLine3	C	According to EMVCo 3DS Spec	According to EMVCo 3DS Spec
54	shipAddrPostCode	C	According to EMVCo 3DS Spec	According to EMVCo 3DS Spec
55	shipAddrState	C	According to EMVCo 3DS Spec	According to EMVCo 3DS Spec
56	transType	C	According to EMVCo 3DS Spec	According to EMVCo 3DS Spec
57	workPhone	C	According to EMVCo 3DS Spec	According to EMVCo 3DS Spec
58	acctID	O	According to EMVCo 3DS Spec	According to EMVCo 3DS Spec
59	acctInfo	O	According to EMVCo 3DS Spec	According to EMVCo 3DS Spec
60	acctType	O	According to EMVCo 3DS Spec	According to EMVCo 3DS Spec
61	merchantRiskIndicator	O	According to EMVCo 3DS Spec	According to EMVCo 3DS Spec
62	messageExtension	O	According to EMVCo 3DS Spec	According to EMVCo 3DS Spec
63	payTokenInd	O	According to EMVCo 3DS Spec	According to EMVCo 3DS Spec
64	purchaseInstalData	O	According to EMVCo 3DS Spec	According to EMVCo 3DS Spec
65	threeDSRequestorAuthenticationInfo	O	According to EMVCo 3DS Spec	According to EMVCo 3DS Spec
66	threeDSRequestorChallengeInd	O	According to EMVCo 3DS Spec	According to EMVCo 3DS Spec
67	threeDSRequestorAuthenticationInd	R	According to EMVCo 3DS Spec	According to EMVCo 3DS Spec
68	threeRIInd	R	According to EMVCo 3DS Spec	According to EMVCo 3DS Spec
69	threeDSRequestorPriorAuthenticationInfo	O	According to EMVCo 3DS Spec	According to EMVCo 3DS Spec
70	threeDSServerRefNumber	R	According to EMVCo 3DS Spec	According to EMVCo 3DS Spec
71	threeDSServerOperatorID	O	According to EMVCo 3DS Spec	According to EMVCo 3DS Spec
72	threeDSServerTransID	R	According to EMVCo 3DS Spec	According to EMVCo 3DS Spec
73	threeDSServerTransID	R	According to EMVCo 3DS Spec	According to EMVCo 3DS Spec
74	threeDSServerURL	R	According to EMVCo 3DS Spec	According to EMVCo 3DS Spec
75	broadInfo	C	According to EMVCo 3DS Spec	According to EMVCo 3DS Spec
76	notificationURL	R	According to EMVCo 3DS Spec	According to EMVCo 3DS Spec

№	Field	Presence	Type/Format	Accepted Value
77	threeDSCompInd	R	According to EMVCo 3DS Spec	According to EMVCo 3DS Spec
78	sdkMaxTimeout	R	According to EMVCo 3DS Spec	According to EMVCo 3DS Spec
79	acsURL	C	According to EMVCo 3DS Spec	According to EMVCo 3DS Spec
80	threeDSRequestorDecMaxTime	C	According to EMVCo 3DS Spec	According to EMVCo 3DS Spec. **Only for 3-D Secure Protocol Version Number 2.2.0.**
81	threeDSRequestorDecReqInd	O	According to EMVCo 3DS Spec	According to EMVCo 3DS Spec. **Only for 3-D Secure Protocol Version Number 2.2.0.**
82	browserJavaScriptEnabled	R	According to EMVCo 3DS Spec	According to EMVCo 3DS Spec. **Only for 3-D Secure Protocol Version Number 2.2.0.**
83	payTokenSource	C	According to EMVCo 3DS Spec	According to EMVCo 3DS Spec. **Only for 3-D Secure Protocol Version Number 2.2.0.**
84	whiteListStatus	O	According to EMVCo 3DS Spec	According to EMVCo 3DS Spec. **Only for 3-D Secure Protocol Version Number 2.2.0.**
85	whiteListStatusSource	C	According to EMVCo 3DS Spec	According to EMVCo 3DS Spec. **Only for 3-D Secure Protocol Version Number 2.2.0.**

Note: Since market or region restriction may vary, any conditional data element with a market or region restriction should be treated as optional for testing purposes.

pArq example Message


```
{
   messageType:"pArq",
   deviceChannel:"01",
   p_messageVersion:"1.0.5",
   sdkAppID:"d3309ce5-4cb9-456b-be21-7e807cd6582e",
   sdkEphemPubKey:{kty : "EC",crv : "P-256",x : "JZ6M72jmi0IR-
cJzKxoIrIVAB3W_M4Walp3vWsdRGo",y :
"93EaQVJEm5i4NBMTBOC2BAeMQ996598B2v_0U7wV4ns"},
   sdkTransID:"2bf8c4dd-56f3-4f7a-a62e-d18b7cd9f0a5",
   sdkReferenceNumber:"Participating UL Test 3DS SDK",
   messageVersion:"2.1.0",
   messageCategory:"01",
   notificationURL: "https://exampleofnotificationurl.com",
   sdkEncData:

"eyJhbGciOiJSU0EtT0FFUCIsImVuYyI6IkExMjhDQkMtSFMyNTYifQ.Mt2iYImdwAaS4t0rWLo3LP
zw3k7UdTIfhUZOkeaOdHJ80g8CPkw6mCbuBw2umHrAln6FNZ7CHiwFLek0FWoTFXJKnfdfPLpLED2k
gcsncaHfjgFN7YDf5551QeKT5hnu02g1ZhzOrdsgMoc2C5HSG6hMSF957hobwwu5_tjDw0KO7PbcWp
YbkmxWURlJRFlNyniM9uIcJAdvxcvXGd3hiayjhOg2xtxCAU51pV9irJ5hDcUJmG8jshOKWkYpcCt9
wm8EWMPBN0sNRbTiOiaOA9hX-duLDu_jdZBTPEZ-
zeEQYMmGnDEm0trONrdqjvuE8aiKtwP8j0EBPfWpfRdF2Q.GcRVvKFetR3ju1vUGRSXhA.Z8ZDAEw_
MciXxi4BuWU6WKYmjM8j6curHoNh3sT-
2JC0ZglsU_JIYh9fP7afHlnnf44ItBhvZtrBiTTBHzg5M62oHtQjmzjHhV9nVR4wyevke73BxGGcMx
SV8vmlIj12y1UvrJnBEH8p2F_Mr_iI7OWOnwQh4EMNyd51HAGKU1lXpEBtr-
qzjR_TEJlUR2_P2JJkq2jpchA2UejLGMQfW_HI-
s0AxBWMRpiD8MDGcxMoUFz0Ch7lm5Q34v_3vjTtVkFWhJIrtjxO42GQib7R-
UvbTnOLrmJQEGrR9Yql3kme_NRi-
T2V2VW7R50zLBzfiJ5jAtMXS5mAfXSa7LqxpksvcnoVyihLTd_CyrQPXrU3uTHnenSZcpsnnQARa-
QfVFt251Kma2L8WOynbTfYEHiLUuchCYm7wV8XHrfOLeXMJZi5ZhCovmT10pyjl3os48gIVT8I0eh1
PC__LKJQxaDuyZv448mi9O8Rd7TKXPrgep-
AAOl1yE0wZzbWXw2h6ni1vGSGg3RtioOXRvYuHArQTJl1TG2ybmcR4oBl2Q7HFlGtULlaZR-
ofqh2xoo0WIV7vbwTBnbrKVn2CesLb4_REUHedKF8r2ty0rEPE5Je-
VX61H3FR2PMuPE37VLn.HFL3DFteaWtUiH9HbRM_XA",
   threeDSRequestorID:"6456",
```

```
threeDSRequestorName:"EMVCo 3DS Test Requestor",
threeDSRequestorURL:"http://ul.com/5060eeab-e499-453e-b0ec-90c539e825e6",
cardExpiryDate:"2212",
acctNumber:"0000000046543778610",
deviceRenderOptions: {
   interface:"3",
   uiType: [
   ]
},
threeDSRequestorAuthenticationInd: "01",
threeDSRequestorAuthenticationInfo: {
   threeDSReqAuthMethod:"01",
   threeDSReqAuthTimestamp:"201710301120",
   threeDSReqAuthData:"00"
},
threeDSServerURL:"https://3dsserver.com/connect",
threeDSRequestorChallengeInd:"01",
threeRIInd:"01",
acctType:"02",
acctInfo: {
   chAccAgeInd:"5",
   chAccDate:"20170101",
   chAccChangeInd:"4",
   chAccChange:"20170101",
   chAccPwChangeInd:"5",
   chAccPwChange:"20170101",
   shipAddressUsageInd:"4",
   shipAddressUsage:"20170101",
   txnActivityDay:"1",
   txnActivityYear:"1",
   provisionAttemptsDay:"0",
   nbPurchaseAccount:"1",
   suspiciousAccActivity:"1",
   shipNameIndicator:"1",
   paymentAccInd:"5",
   paymentAccAge:"20170101"
},
acctID:"EMVCo 3DS Test Account 000000001",
merchantRiskIndicator: {
   shipIndicator:"01",
   deliveryTimeframe:"2",
   deliveryEmailAddress:"example@example.com",
   reorderItemsInd:"1",
   preOrderPurchaseInd:"1",
   preOrderDate:"20300101",
   giftCardAmount:"1",
   giftCardCurr:"840",
   giftCardCount:"01"
},
shipAddrCity:"City Name",
shipAddrPostCode:"Postal Code",
purchaseDate:"20171030112014",
purchaseExponent:"2",
acquirerMerchantID:"9876543210001",
merchantName:"Ticket Service",
billAddrCountry:"840",
purchaseAmount:"2334",
mcc:"7922",
billAddrLine2:"Address Line 2",
workPhone: {
   cc:"123",
   subscriber:"123456789"
},
homePhone: {
   cc:"123",
```

```
      subscriber:"123456789"
   },
   shipAddrLine1:"Address Line 1",
   shipAddrLine2:"Address Line 2",
   mobilePhone: {
      cc:"123",
      subscriber:"123456789"
   },
   billAddrLine1:"Address Line 1",
   billAddrPostCode:"Postal Code",
   merchantCountryCode:"840",
   billAddrCity:"City Name",
   addrMatch:"Y",
   email:"example@example.com",
   transType:"01",
   billAddrLine3:"Address Line 3",
   cardholderName:"Frictionless One",
   shipAddrState:"AZ",
   purchaseCurrency:"840",
   shipAddrLine3:"Address Line 3",
   acquirerBIN:"000000999",
   billAddrState:"AZ",
   shipAddrCountry:"840",
   sdkMaxTimeout: "05"
}
```

A.2.2 Proprietary Authentication Response (pArs)

№	Field	Presence	Type/Format	Accepted Value
1.	threeDSServerTransID	R	According to EMVCo 3DS Spec	According to EMVCo 3DS Spec
2.	p_messageVersion	R	1.0.5	Starting value: 1.0.0
3.	<removed>	<removed>	<removed>	<removed>
4.	messageType	R	According to EMVCo 3DS Spec	pArs
5.	messageVersion	R	According to EMVCo 3DS Spec	According to EMVCo 3DS Spec
6.	transStatus	R	According to EMVCo 3DS Spec	According to EMVCo 3DS Spec
7.	dsReferenceNumber	R	According to EMVCo 3DS Spec	According to EMVCo 3DS Spec
8.	acsReferenceNumber	R	According to EMVCo 3DS Spec	According to EMVCo 3DS Spec
9.	acsTransID	R	According to EMVCo 3DS Spec	According to EMVCo 3DS Spec
10.	dsTransID	R	According to EMVCo 3DS Spec	According to EMVCo 3DS Spec
11.	authenticationValue	C	According to EMVCo 3DS Spec	According to EMVCo 3DS Spec
12.	<removed>	<removed>	<removed>	<removed>
13.	acsRenderingType	C	According to EMVCo 3DS Spec	According to EMVCo 3DS Spec
14.	acsOperatorID	O	According to EMVCo 3DS Spec	According to EMVCo 3DS Spec
15.	acsSignedContent	C	According to EMVCo 3DS Spec	According to EMVCo 3DS Spec
16.	acsURL	C	According to EMVCo 3DS Spec	According to EMVCo 3DS Spec
17.	authenticationType	C	According to EMVCo 3DS Spec	According to EMVCo 3DS Spec
18.	acsChallengeMandated	C	According to EMVCo 3DS Spec	According to EMVCo 3DS Spec
19.	eci	C	According to EMVCo 3DS Spec	According to EMVCo 3DS Spec
20.	messageExtension	C	According to EMVCo 3DS Spec	According to EMVCo 3DS Spec
21.	<removed>	<removed>	<removed>	<removed>
22.	sdkTransID	R	According to EMVCo 3DS Spec	According to EMVCo 3DS Spec
23.	transStatusReason	C	According to EMVCo 3DS Spec	According to EMVCo 3DS Spec
24.	cardholderInfo	O C (**for v2.2.0**)	According to EMVCo 3DS Spec	According to EMVCo 3DS Spec
25.	broadInfo	C	According to EMVCo 3DS Spec	According to EMVCo 3DS Spec
26	acsDecConInd	C	According to EMVCo 3DS Spec	According to EMVCo 3DS Spec. **Only for 3-D Secure Protocol Version Number 2.2.0.**
27	whiteListStatus	O	According to EMVCo 3DS Spec	According to EMVCo 3DS Spec. **Only for 3-D Secure Protocol Version Number 2.2.0.**
28	whiteListStatusSource	C	According to EMVCo 3DS Spec	According to EMVCo 3DS Spec. **Only for 3-D Secure Protocol Version Number 2.2.0.**

pArs example message


```
{
   acsTransID:"dfb6e03d-c3ba-4271-915a-95886331fc31",
   p_messageVersion:"1.0.5",
   eci:"00",
   dsReferenceNumber:"DS",
   acsReferenceNumber:"ACS",
   messageType:"pArs",
   dsTransID:"f56b9153-1624-4ac4-ba94-39a187110868",
   messageVersion:"2.1.0",
```

```
    sdkTransID:"2bf8c4dd-56f3-4f7a-a62e-d18b7cd9f0a5",
    authenticationValue:"VUwgQUNTIFNpbXVsYXRvciA1NDg=",
    transStatus:"Y",
    threeDSServerTransID:"cedec51a-2163-470d-b236-b295638b12e1"
}
```

A.2.3 Proprietary Get Challenge Request (pGcq)

№	Field	Presence	Type/Format	Accepted Value
1.	messageType	R	According to EMVCo 3DS Spec	pGcq
2.	p_messageVersion	R	1.0.5	Starting value: 1.0.0
3.	messageVersion	R	According to EMVCo 3DS Spec	According to EMVCo 3DS Spec
4.	threeDSServerTransID	R	According to EMVCo 3DS Spec	According to EMVCo 3DS Spec
5.	acsTransID	R	According to EMVCo 3DS Spec	According to EMVCo 3DS Spec
6.	threeDSSessionData	O	According to EMVCo 3DS Spec	According to EMVCo 3DS Spec
7.	challengeWindowSize	R	According to EMVCo 3DS Spec	According to EMVCo 3DS Spec

pGcq example message

```
{
  "messageType": "pGcq",
  "p_messageVersion": "1.0.5",
  "messageVersion": "2.1.0",
  "threeDSServerTransID": "8a880dc0-d2d2-4067-bcb1-b08d1690b26e",
  "acsTransID": "d7c1ee99-9478-44a6-b1f2-391e29c6b340",
  "threeDSSessionData": "TG9yZW0gaXBzdW0=",
  "challengeWindowSize": "05"
}
```

A.2.4 Proprietary Get Challenge Response (pGcs)

№	Field	Presence	Type/Format	Accepted Value
1.	messageType	R	According to EMVCo 3DS Spec	pGcs
2.	p_messageVersion	R	1.0.5	Starting value: 1.0.0
3.	messageVersion	R	According to EMVCo 3DS Spec	According to EMVCo 3DS Spec
4.	htmlCreq	R	Base64 encoded Creq, According to EMVCo 3DS Spec	According to EMVCo 3DS Spec

pGcs example message

```
{
  "messageType": "pGcs",
  "p_messageVersion": "1.0.5",
  "messageVersion": "2.1.0",
  "htmlCreq": "<form action=\'https://acs.mybank.example\'
method=\'post\'><input type=\'hidden\' name=\'creq\'
value=\'ewogICJ0aHJlZURTU2VydmVyVHJhbnNJRCIgOiAiOGE4ODBkYzAtZDJkMi00MDY3LWJjYj
EtYjA4ZDE2OTBiMjZlIiwKICAiYWNzVHJhbnNJRCIgOiAiZDdjMWVlOTktOTQ3OC00NGE2LWIxZjIt
MzkxZTI5YzZiMzQwIiwKICAibWVzc2FnZVR5cGUiIDogIkNSZXEiLAogICJtZXNzYWdlVmVyc2lvbi
IgOiAiMi4wLjAiCn0=\' /><input type=\'hidden\' name=\'threeDSSessionData\'
```

```
value=\'b\'TG9yZW0gaXBzdW0=\'\' /></form>"
}
```

A.2.5 Proprietary Information Request (pIrq)

№	Field	Presence	Type/Format	Accepted Value
1.	messageType	R	Length: 4 character JSON Data Type: String	pIrq
2.	messageVersion	R	According to EMVCo 3DS Spec	According to EMVCo 3DS Spec
3.	p_messageVersion	R	1.0.5	Starting value: 1.0.0
4.	acsTransID	R	According to EMVCo 3DS Spec	According to EMVCo 3DS Spec
5.	p_formValues_BRW	C	JSON Object	Proprietary Browser Form Values, see Table 6. Only applicable to 02-BRW Device Channel.
6.	p_formValues_APP	C	JSON Object	Proprietary Application Form Values, see Table 7

Table 6 Proprietary Browser Form Value

Data Element/Field Name	Description	Format	Message Inclusion
Form Action Field Name: `action`	URL to post the form data to. Via this URL the ACS receives the challenge data entered by the cardholder in the browser.	JSON Data Type: String Contains fully qualified URL to reach the ACS.	pIrq = R
Correct Form Data Field Name: `correctFormData`	Correct challenge data as entered by the Cardholder. Full result of the HTML form submission as expected by the ACS.	JSON Data Type: String	pIrq = R
Incorrect Form Data Field Name: `incorrectFormData`	Incorrect challenge data as entered by the Cardholder. Full result of the HTML form submission as expected by the ACS.	JSON Data Type: String	pIrq = R
Cancel Form Data Field Name: `cancelFormData`	Data sent to the ACS in case the Cardholder cancels the challenge flow. Full result of the HTML form submission as expected by the ACS.	JSON Data Type: String	pIrq = R

Table 7 Proprietary Application Form Values

Data Element/Field Name	Description	Format	Message Inclusion
Correct Challenge Data Field Name:	Correct challenge data as entered by the Cardholder.	JSON Data Type: String	pIrq = R

correctChallengeData	For HTML UI this should be the full result of the HTML form submission as expected by the ACS.		
Incorrect Challenge Data Field Name: incorrectChallengeData	Incorrect challenge data as entered by the Cardholder. For HTML UI this should be the full result of the HTML form submission as expected by the ACS.	JSON Data Type: String	pIrq = R

A.2.5.1 Browser pIrq Example Message

```
{
  "messageType": "pIrq",
  "messageVersion": "2.1.0",
  "p_messageVersion": "1.0.5",

  "acsTransID": "8a880dc0-d2d2-4067-bcb1-b08d1690b26e",

  "p_formValues_BRW": {
    "action": "<ACS Challenge URL>",
    "correctFormData": "pet=Spot&otp=1234",
    "incorrectFormData": "pet=Spot&otp=0000",
    "cancelFormData": "cancel=true"
  }
/* These values will be submitted as POST data (application/x-www-form-
urlencoded); the values are assumed to have been formatted correctly.
In case of OOB, correctFormData is used to submit the form. */
  }
}
```

A.2.5.2 App, Native UI, Text pIrq Example Message

```
{
  "messageType": "pIrq",
  "messageVersion": "2.1.0",
  "p_messageVersion": "1.0.5",
  "acsTransID": "8a880dc0-d2d2-4067-bcb1-b08d1690b26e",

   "p_formValues_APP" : {
     "correctChallengeData": "1234",
     "incorrectChallengeData": "12345"
/* These correct or incorrect values will be submitted in the
challengeDataEntry or challengeHTMLDataEntry fields inside the Creq */
  }
}
```

A.2.6 Proprietary Information Response (pIrs)

№	Field	Presence	Type/Format	Accepted Value
1.	messageType	R	Length: 4 character JSON Data Type: String	pIrs
2.	messageVersion	R	According to EMVCo 3DS Spec	According to EMVCo 3DS Spec

3.	p_messageVersion	R	1.0.5	Starting value: 1.0.0
4.	acsTransID	R	According to EMVCo 3DS Spec	According to EMVCo 3DS Spec

pIrs example message

```
{
  "messageType": "pIrs",
  "messageVersion": "2.1.0",
  "p_messageVersion": "1.0.5",
  "acsTransID": "8a880dc0-d2d2-4067-bcb1-b08d1690b26e"
}
```

A.2.7 Proprietary Out-of-Band Request (pOrq)

№	Field	Presence	Type/Format	Accepted Value
1.	messageType	R	Length: 4 character JSON Data Type: String	pOrq
2.	messageVersion	R	According to EMVCo 3DS Spec	According to EMVCo 3DS Spec
3.	p_messageVersion	R	1.0.5	Starting value: 1.0.0
4.	threeDSServerTransId	R	According to EMVCo 3DS Spec	According to EMVCo 3DS Spec
5.	acsTransID	R	According to EMVCo 3DS Spec	According to EMVCo 3DS Spec

pOrq example message

```
{
  "messageType": "pOrq",
  "messageVersion": "2.1.0",
  "p_messageVersion": "1.0.5",
  "threeDSServerTransID": "3h539pg6-j1i2-5090-pcp1-d64u1812j45r",
  "acsTransID": "8a880dc0-d2d2-4067-bcb1-b08d1690b26e"
}
```

A.2.8 Proprietary Out-of-Band Response (pOrs)

№	Field	Presence	Type/Format	Accepted Value
1.	messageType	R	According to EMVCo 3DS Spec	pOrs
2.	messageVersion	R	According to EMVCo 3DS Spec	According to EMVCo 3DS Spec
3.	p_messageVersion	R	1.0.5	Starting value: 1.0.0
4.	p_isOobSuccessful	R	Boolean	true OR false

pOrs example message

```
{
  "messageType": "pOrs",
  "messageVersion": "2.1.0",
  "p_messageVersion": "1.0.5",
  "p_isOobSuccessful": true
}
```

A.2.9 Proprietary Preparation Request (pPrq)

№	Field	Presence	Type/Format	Accepted Value
1.	messageType	R	According to EMVCo 3DS Spec	pPrq
2.	p_messageVersion	R	1.0.5	Starting value: 1.0.0
3.	messageVersion	R	According to EMVCo 3DS Spec	According to EMVCo 3DS Spec
4.	threeDSRequestorID	R	According to EMVCo 3DS Spec	According to EMVCo 3DS Spec
5.	threeDSServerTransID	R	According to EMVCo 3DS Spec	According to EMVCo 3DS Spec
6.	threeDSRequestorURL	R	According to EMVCo 3DS Spec	According to EMVCo 3DS Spec
7.	messageExtension	O	According to EMVCo 3DS Spec	According to EMVCo 3DS Spec

pPrq example message

```
{
  "messageType": "pPrq",
  "messageVersion": "2.1.0",
  "p_messageVersion": "1.0.5",
  "threeDSRequestorID": "456",
  "threeDSServerTransID": "b02a9428-c46a-11e7-abc4-cec278b6b50a",
  "threeDSRequestorURL": " http://ul.com/5060eeab-e499-453e-b0ec-90c539e825e6"
  "messageExtension": [{
    "name": "testExtensionNonCriticalField",
    "id": "ID3",
    "criticalityIndicator": false,
    "data": "This is a test non-critical message extension"
  }
]

}
```

A.2.10 Proprietary Preparation Response (pPrs)

№	Field	Presence	Type/Format	Accepted Value
1.	messageType	R	Length: 4 character JSON Data Type: String	pPrs
2.	messageVersion	R	According to EMVCo 3DS Spec	According to EMVCo 3DS Spec
3.	p_messageVersion	R	1.0.5	Starting value: 1.0.0
4.	p_completed	R	JSON Data Type: Boolean	True
5	messageExtension	O	According to EMVCo 3DS Spec	According to EMVCo 3DS Spec

pPrs example message

```
{
  "messageType": "pPrs",
  "messageVersion": "2.1.0",
  "p_messageVersion": "1.0.5",
  "p_completed": true
}
```

A.3 Internal proprietary messages

This section contains a detailed list of all internal proprietary messages used internally by the UL 3DS Self Test Platform. These messages should not be implemented by other parties or Product Providers. These are used for internal communication between UL components, and could be visible in the UL 3DS Self Test Platform interface.

A.3.1 Proprietary SDK Information Request (pSrq)

№	Field	Presence	Type/Format	Accepted Value
1.	messageType	R	Length: 4 character JSON Data Type: String	pSrq
2.	messageVersion	R	According to EMVCo 3DS Spec	According to EMVCo 3DS Spec
3.	p_messageVersion	R	1.0.5	Starting value: 1.0.0
4.	threeDSServerTransID	R	According to EMVCo 3DS Spec	According to EMVCo 3DS Spec
5.	p_isTransactionCompleted	R	String	"true" or "false"
6.	p_challengeCancel	O	Length: 2 characters; numeric	01: cancel the transaction
7.	p_challengeTimeout	O	Boolean	True
8.	p_protocolErrorCode	O	To be added	To be added
9.	p_protocolErrorMessage	O	To be added	To be added
10.	p_runtimeErrorCode	O	To be added	To be added
11.	p_runtimeErrorMessage	O	To be added	To be added

pSrq example message

Challenge Completed Scenario (either Y or N)
Called when the challenge process (that is, the transaction) is completed. When a transaction is completed, a transaction status shall be available.
So the pSrq would be:
`{ messageType=pSrq, messageVersion=2.1.0, p_isTransactionCompleted= "true", p_messageVersion=1.0.5, threeDSServerTransID=7d20e188-e05d-40c1-b74c-8d53106f1acb }`
or
`{ messageType=pSrq, messageVersion=2.1.0, p_isTransactionCompleted= "false", p_messageVersion=1.0.5, threeDSServerTransID=7d20e188-e05d-40c1-b74c-8d53106f1acb }`

Challenge Cancel Scenario
Called when the Cardholder selects the option to cancel the transaction on the challenge screen.
So the pSrq would be:

```
{ messageType=pSrq, messageVersion=2.1.0, p_challengeCancel=01,
p_isTransactionCompleted=false, p_messageVersion=1.0.5,
threeDSServerTransID=2b03a052-524b-491c-9d40-c315d5263c2c }
```

Challenge Timeout Scenario
Called when the challenge process reaches or exceeds the timeout interval that is specified during the doChallenge call on the 3DS SDK.
So the pSrq would be:

```
{ messageType=pSrq, messageVersion=2.1.0, p_challengeTimeout=true,
p_isTransactionCompleted=false, p_messageVersion=1.0.5,
threeDSServerTransID=2b03a052-524b-491c-9d40-c315d5263c2c }
```

Challenge Protocol Error Scenario
Called when the 3DS SDK receives an EMV 3-D Secure specifications[1]-defined error message from the ACS.
So the pSrq would be:

```
{ messageType=pSrq, messageVersion=2.1.0, p_protocolErrorCode=999,
p_protocolErrorMessage=ACS Exception, p_isTransactionCompleted=false,
p_messageVersion=1.0.5, threeDSServerTransID=2b03a052-524b-491c-9d40-
c315d5263c2c }
```

Challenge Runtime Error Scenario
Called when the 3DS SDK encounters errors during the challenge process. These errors include all errors except those covered by the protocolError method.
So the pSrq would be:

```
{ messageType=pSrq, messageVersion=2.1.0, p_runtimeErrorCode=999,
p_runtimeErrorMessage=Runtime Exception, p_isTransactionCompleted=false,
p_messageVersion=1.0.5, threeDSServerTransID=2b03a052-524b-491c-9d40-
c315d5263c2c }
```

A.3.2 Proprietary SDK Information Response (pSrs)

№	Field	Presence	Type/Format	Accepted Value
1.	messageType	R	Length: 4 character JSON Data Type: String	pSrs
2.	messageVersion	R	According to EMVCo 3DS Spec	According to EMVCo 3DS Spec
3.	p_messageVersion	R	1.0.5	Starting value: 1.0.0

pSrs example message

```
{

{"messageType":"pSrs","messageVersion":"2.1.0","p_messageVersion":"1.0.5"}

}
```

A.4 Browser flow HTML message examples

A.4.1 CReq HTTP POST form

This challenge request will be sent in an HTTP POST form. The form will contain the Base64url encoded CReq and optionally a Base64url encoded threeDSSessionData field as per the EMVCo specifications. An example of this in POST form received by the ACS is shown here –

```
POST /acs/threeDSBrowserChallenge / HTTP/1.1
Host: acs.mybank.example
Content-Type: application/x-www-form-urlencoded
creq=eyJ0aHJlZURTU2VydmVyVHJhbnNJRCI6IjJmZmNjYTRiLTRjZDYtNDBkYy1iYjI5LTg1YmJhZ
WMyMTFhNiIsImFjc1RyYW5zSUQiOiJmODE4ZTBmMi01OTUzLTQwYjgtOGM5ZC01MzMxNzA1YTUwODA
iLCJtZXNzYWdlVHlwZSI6IkNSZXEiLCJtZXNzYWdlVmVyc2lvbiI6IjIuMS4wIiwiY2hhbGxlbmdlV
2luZG93U2l6ZSI6IjAyIn0%3D&threeDSSessionData=VGhpcyBpcyBteSBzZXNzaW9uIGRhdGEgM
TIzNDU2Nzg5MA%3D
```

A.4.2 Final CRes HTML page

The UL Test Environment requires that this HTML page contain a single form element with the CRes and optionally the threeDSSessionData. The data elements need to be Base64url encoded as per Table A.3 of the Specification.

An example of the HTML page with the CRes and threeDSSessionData expected to be returned to the UL Platform is shown below –

```
<html>
<form action="https://www.mymerchant.example/3ds-complete" method="post">
<input name="cres" value="
ewogICJhY3NUcmFuc0lEIiA6ICJmODE4ZTBmMi01OTUzLTQwYjgtOGM5ZC01MzMxNzA1YTUwODAiLA
ogICJtZXNzYWdlVHlwZSIgOiAiQ1JlcyIsCiAgIm1lc3NhZ2VWZXJzaW9uIiA6ICIyLjEuMCIsCiAg
InRocmVlRFNTZXJ2ZXJUcmFuc0lEIiA6ICIyZmZjY2E0Yi00Y2Q2LTQwZGMtYmIyOS04NWJiYWVjMj
ExYTYiLAogICJ0cmFuc1N0YXR1cyIgOiAiTiIKfQ">
<input name="threeDSSessionData" value="
VGhpcyBpcyBteSBzZXNzaW9uIGRhdGEgMTIzNDU2Nzg5MA">
</form>
</html>
```

A.5 DS Public Keys

To successfully execute the test cases, the 3DS SDK needs to be configured with the DS Public Keys, each identified by the Directory Server ID (which is the Payment Systems RID) listed in the following table.

Note: The DS Public Keys listed here are in PEM format.

Table 8 DS public keys

Directory Server ID (RID)	DS Public Key	Algorithm
F0 00 00 00 00	MIIBIjANBgkqhkiG9w0BAQEFAAOCAQ8AMIIBCgKCAQEAr/O0BfXWngO9OJDBsqdR5U2h28jrX6Y+LlblTBaYeT2tW7+ca3YzTFXA8duVUwdlWxl3JZCOOeL1feVP6g0TNOHVCkCnirVDLkcozod4aSkNvx+929aDr1ithqhruf0skBc2sMZGBBCNpso6XGzyAf2uZ2+9DvXoKIUYgcr7PQmL2Y0awyQN7KCRcusaotYNz2mOPrL/hAv6hTexkNrQKzFcPwCuc6kN6aNjD+p2CJ51/5p02SNS70nPOmwmg63j6f3n7xVykQ56kNc1l5B5xOpeHJmqk3+hyF1dF/47rQmMFicN41QSvZ5AZJKgWlIn2VQROMkEHkF9ZBRLx1nFTwIDAQAB	RSA
F0 00 00 00 01	MFkwEwYHKoZIzj0CAQYIKoZIzj0DAQcDQgAEYktbLuAv0v52erE5LPscomKaOmQsvevxzOyn9k4sF1hqpBc5kUygzxA9Jl0R/2dTuk8ka7UCujk36xeUsLVpWA==	EC

The test cases verify that the 3DS SDK selects the correct DS Public Key based on the Directory Server ID provided by the 3DS Requestor App and is able to encrypt the Device Information using either RSA or EC cryptography.

A.6 Proprietary Error Component Values

Next to the Error Component values for ACS (A), DS (D), 3DS Server (S) and 3DS SDK (C) defined in the EMV® 3-D Secure Protocol and Core Functions specification, the test platform uses some proprietary values for the internal components introduced as part of the Test Harness. These values are:

- O for the OOB Server used during out-of-band challenge flows.
- F for the Cardholder which receives the proprietary SDK Information Request from the UL Reference App.
- I for the Challenge Information Server (CIS) used during the challenge flows.

In exceptional cases you may observe these values in Erro messages sent by the platform to the System Under Test.