



EMV®

3-D Secure

Protocol and Core Functions Specification

Version 2.3.1.0

August 2022

Legal Notice

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

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

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

Revision Log

The following table lists the version history for the *EMV® 3-D Secure Protocol and Core Functions Specification*. *EMV® Specification Bulletins* provide the detailed updates made with each specification release.

Version	Release Date	Associated Specification Bulletins
2.0.0	October 2016	<ul style="list-style-type: none">• SB 190: 3-D Secure Requirement Numbering Scheme and Error Processing• SB 196: 3-D Secure Updates, Clarifications & Errata
2.1.0	October 2017	<ul style="list-style-type: none">• SB 204v4: 3-D Secure Updates, Clarifications & Errata• SB 214v1: EMV® 3-D Secure Updates, Clarifications & Errata
2.2.0	December 2018	<ul style="list-style-type: none">• SB 207: 3-D Secure Updates, Clarifications & Errata
2.3.0.0	September 2021	<ul style="list-style-type: none">• SB 227v1: EMV® 3-D Secure Key Features v2.3.0.0
2.3.1.0	August 2022	<ul style="list-style-type: none">• SB 256: EMV® 3-D Secure Protocol and Core Functions Specification v2.3.1.0

Contents

Legal Notice	2
Revision Log	3
Contents	4
Figures.....	13
Tables	15
1 Introduction	17
1.1 Purpose.....	17
1.2 Audience	17
1.3 Normative References.....	18
1.4 Acknowledgement.....	20
1.5 Definitions	21
1.6 Abbreviations.....	31
1.7 3-D Secure Protocol Version Number.....	32
1.8 Supporting Documentation	32
1.9 Terminology and Conventions	33
1.10 Constraints	34
2 EMV 3-D Secure Overview	35
2.1 Acquirer Domain.....	36
2.1.1 3DS Requestor Environment	36
2.1.1.1 3DS Requestor	36
2.1.1.2 3DS Client	36
2.1.1.3 3DS Server.....	37
2.1.2 3DS Integrator (3DS Server and 3DS Client)	37
2.1.3 Acquirer (Payment Authorisation)	37
2.2 Interoperability Domain.....	38
2.2.1 Directory Server	38
2.2.2 Directory Server Certificate Authority	38
2.2.3 Authorisation System (Payment Authentication)	39
2.3 Issuer Domain	39
2.3.1 Cardholder.....	39
2.3.2 Consumer Device	39
2.3.3 Issuer.....	39
2.3.4 Access Control Server	40
2.4 3-D Secure Messages	40

2.4.1	Authentication Request Message (AReq)	40
2.4.2	Authentication Response Message (ARes).....	40
2.4.3	Challenge Request Message (CReq).....	40
2.4.4	Challenge Response Message (CRes)	40
2.4.5	Results Request Message (RReq).....	41
2.4.6	Results Response Message (RRes)	41
2.4.7	Preparation Request Message (PReq).....	41
2.4.8	Preparation Response Message (PRes)	41
2.4.9	Operation Request Message (OReq).....	41
2.4.10	Operation Response Message (ORes)	41
2.4.11	Error Message	41
2.5	Authentication Flows	42
2.5.1	Frictionless Flow	42
2.5.2	Challenge Flow	42
2.5.3	Processing EMV Payment Tokens.....	42
2.6	Frictionless Flow Outline	43
2.6.1	3DS Requestor Environment—App-based	44
2.6.2	3DS Requestor Environment—Browser-based	45
2.6.3	3DS Requestor Environment—3RI	45
2.7	Challenge Flow Outline	47
3	EMV 3-D Secure Authentication Flow Requirements	49
3.1	App-based Requirements	50
Step 1:	The Cardholder	51
Step 2:	The 3DS Requestor App.....	51
Step 3:	The 3DS SDK.....	51
Step 4:	The 3DS Requestor Environment	51
Step 5:	The 3DS Server.....	52
Step 6:	The DS	53
Step 7:	The ACS.....	55
Step 8:	The DS	57
Step 9:	The 3DS Server.....	58
Step 10	The 3DS Requestor App.....	59
Step 11:	The 3DS Requestor Environment	59
Step 12:	The ACS.....	59
Step 13:	The ACS.....	60
Step 14:	The 3DS SDK.....	60
Step 15:	The Cardholder Interaction with the 3DS SDK.....	60
Step 16:	The 3DS SDK.....	60

Step 17: The ACS.....	61
Step 18: The ACS.....	62
Step 19: The DS	62
Step 20 The 3DS Server.....	63
Step 21 The DS	63
Step 22 The ACS.....	63
Step 23 The ACS.....	63
Step 24 The 3DS Requestor Environment.....	64
Step 25 The 3DS Requestor App.....	64
3.2 Challenge Flow with OOB Authentication Requirements	65
3.2.1 OOB Requirements.....	66
Step 15 The Cardholder Interaction with the 3DS SDK.....	66
3.2.2 OOB Automatic Switching Features.....	66
Step 13: The ACS.....	66
Step 14: The 3DS SDK.....	67
Step 15: The Cardholder Interaction with the 3DS SDK.....	67
3.2.2.1 OOB App URL Requirements.....	67
3.2.2.2 3DS Requestor App URL.....	67
3.3 Browser-based Requirements	69
Step 1: The Cardholder	69
Step 2: The 3DS Server/3DS Requestor.....	69
Step 3 The 3DS Requestor Environment.....	70
Step 4 Browser and the ACS	70
Step 5: The 3DS Requestor Environment.....	70
Step 6 The 3DS Server.....	71
Step 7 The DS	72
Step 8 The ACS.....	73
Step 9 The DS	75
Step 10 The 3DS Server.....	76
Step 11 The ACS.....	77
Step 12 The ACS and Browser.....	78
Step 13 The Cardholder	78
Step 14 The Browser.....	78
Step 15 The ACS.....	78
Step 16 The ACS.....	79
Step 17 The DS	80

Step 18 The 3DS Server.....	80
Step 19 The DS	80
Step 20 The ACS.....	81
Step 21 The ACS.....	81
Step 22 The 3DS Requestor Environment.....	81
3.4 3RI-based Requirements.....	82
Step 1: The 3DS Requestor.....	83
Step 2 The 3DS Server.....	83
Step 3 The DS	84
Step 4 The ACS.....	85
Step 5 The DS	86
Step 6 The 3DS Server.....	87
Step 7 The ACS.....	87
Step 8 The ACS.....	88
Step 9 The DS	88
Step 10 and Step 11 The 3DS Server	88
Step 12 The DS	89
Step 13 The ACS.....	89
3.5 SPC-based Authentication Requirements.....	89
3.5.1 3DS Requestor Initiates SPC Authentication.....	89
Step 10 The 3DS Server.....	91
Step 10a The 3DS Requestor	91
Step 10b The Cardholder.....	91
Step 10c The 3DS Requestor	91
Step 10d The 3DS Requestor	91
Step 10e The 3DS Server	91
Step 10f The DS	92
Step 10g The ACS.....	92
Step 10h The DS	92
Step 10i The 3DS Server	92
3.5.2 The ACS Initiates SPC Authentication	92
4 EMV 3-D Secure User Interface Templates, Requirements and Guidelines	94
4.1 3-D Secure User Interface Templates.....	94
4.2 App-based User Interface Overview	98
4.2.1 Processing Screen Requirements	99
4.2.1.1 3DS SDK/3DS Requestor App.....	100

4.2.2	Native UI Display Requirements	104
4.2.2.1	3DS SDK/ACS.....	105
4.2.3	Native UI Templates	106
4.2.4	Native UI Message Exchange Requirements	123
4.2.4.1	3DS SDK.....	123
4.2.4.2	ACS.....	123
4.2.4.3	3DS SDK.....	123
4.2.5	HTML UI Display Requirements.....	124
4.2.5.1	3DS SDK/ACS.....	125
4.2.6	HTML UI Templates.....	125
4.2.6.1	HTML Other UI Templates.....	133
4.2.7	HTML Message Exchange Requirements.....	135
4.2.7.1	3DS SDK.....	135
4.2.7.2	ACS.....	135
4.2.7.3	3DS SDK.....	136
4.3	Browser-based User Interface Overview	136
4.3.1	Processing Screen Requirements.....	137
4.3.1.1	3DS Requestor Website	137
4.3.1.2	ACS.....	137
4.3.2	Browser Display Requirements	138
4.3.2.1	ACS.....	138
4.3.3	Browser UI Templates	139
4.4	3RI Considerations	144
5	EMV 3-D Secure Message Handling Requirements	145
5.1	General Message Handling	145
5.1.1	HTTP POST.....	145
5.1.2	HTTP Header—Content-Type.....	145
5.1.3	Base64/Base64url Encoding.....	146
5.1.4	Protocol and Message Version Numbers	146
5.1.5	Data Version Numbers.....	147
5.1.6	Message Parsing	147
5.1.7	Message Content Validation	148
5.2	Partial System Outages	149
5.3	3-D Secure Component Availability	150
5.4	Error Codes	150
5.5	Timeouts	150
5.5.1	Transaction Timeouts	150
5.5.2	Read Timeouts	152

5.5.2.1 AReq/ARes Message Timeouts.....	152
5.5.2.2 RReq/RRes Message Timeouts	154
5.6 PReq/PRes Message Handling Requirements	154
5.7 App/SDK-based Message Handling	157
5.7.1 App-based CReq/CRes Message Handling	157
5.8 Browser-based Message Handling	158
5.8.1 3DS Method Handling	158
5.8.1.1 Recent Prior 3DS Method Call Does Not Exist	158
5.8.1.2 Recent Prior 3DS Method Call Does Exist.....	159
5.8.2 Browser Challenge iframe Requirements.....	159
5.9 Message Error Handling	160
5.9.1 DS AReq Message Error Handling.....	161
5.9.2 ACS AReq Message Error Handling	161
5.9.3 DS ARes Message Error Handling	161
5.9.3.1 Message in Error	162
5.9.3.2 Error Message Received	162
5.9.4 3DS Server ARes Message Error Handling.....	162
5.9.5 ACS CReq Message Error Handling—01-APP	163
5.9.5.1 Message in Error	164
5.9.6 ACS CReq Message Error Handling—02-BRW	164
5.9.7 3DS SDK CRes Message Error Handling	165
5.9.8 DS RReq Message Error Handling	165
5.9.8.1 Message in Error	166
5.9.9 3DS Server RReq Message Error Handling	166
5.9.10 DS RRes Message Error Handling.....	167
5.9.11 ACS RRes Message Error Handling—01-APP.....	167
5.9.12 ACS RRes Message Error Handling—02-BRW	168
5.9.13 ACS RRes Message Error Handling—03-3RI	169
5.10 UTC Date and Time.....	169
5.11 OReq/ORes Message Handling Requirements.....	170
6 EMV 3-D Secure Security Requirements	172
6.1 Links.....	173
6.1.1 Link a: Consumer Device—3DS Requestor	173
6.1.2 Link b: 3DS Server—DS	174
6.1.2.1 For AReq/ARes	174
6.1.2.2 For RReq/RRes	174
6.1.3 Link c: DS—ACS.....	174
6.1.3.1 For AReq/ARes	174

6.1.3.2 For RReq/RRes	175
6.1.4 Link d: 3DS Client—ACS	175
6.1.4.1 For App-based CReq/CRes	175
6.1.4.2 For Browser-based CReq/CRes	175
6.1.5 Link e: 3DS Integrator—Acquirer (Payment Authentication only)	176
6.1.6 Link f: Acquirer—Payment System (Payment Authentication only).....	176
6.1.7 Link g: Payment System—Issuer (Payment Authentication only)	176
6.1.8 Link h: Browser—ACS (for 3DS Method)	176
6.2 Security Functions	177
6.2.1 Function H: Authenticity of the 3DS SDK	177
6.2.2 Function I: 3DS SDK Device Information Encryption and Split-SDK Server Signature to DS	177
6.2.2.1 3DS SDK Device Information Encryption	178
6.2.2.2 DS Device Information Decryption	179
6.2.2.3 Split-SDK Server Signature	179
6.2.2.4 DS Verification of Split-SDK Server Signed Content.....	180
6.2.3 Function J: 3DS SDK—ACS Secure Channel Set-Up	180
6.2.3.1 3DS SDK Preparation for Secure Channel	180
6.2.3.2 ACS Secure Channel Setup	180
6.2.3.3 3DS SDK Secure Channel Setup	182
6.2.4 Function K: 3DS SDK—ACS (CReq, CRes).....	182
6.2.4.1 3DS SDK—CReq	182
6.2.4.2 3DS SDK—CRes.....	183
6.2.4.3 ACS—CReq	183
6.2.4.4 ACS—CRes	183
Annex A 3-D Secure Data Elements.....	185
A.1 Missing Required Fields	187
A.2 Field Edit Criteria.....	187
A.3 Encryption of AReq Data	187
A.4 EMV 3-D Secure Data Elements	188
A.5 Device Information—01-APP Only.....	306
A.6 Browser Information—02-BRW Only	306
A.7 3DS Method Data.....	307
3DS Method Data Examples.....	309
A.8 Browser CReq and CRes POST	309
Browser CReq - CRes Data Examples.....	311
A.9 Error Code, Error Description, and Error Detail.....	313

A.10 Excluded ISO Currency and Country Code Values.....	319
A.11 Card Range Data	320
Card Range Data Example	324
DS URL List Data Example.....	326
A.11.1 Supported Message Extension Data Element.....	327
Supported Message Extension Data Example	328
A.12 Message Extension Data.....	329
A.12.1 Message Extension Attributes.....	331
A.12.2 Identification.....	332
A.12.3 Criticality	332
A.13 3DS Requestor Risk Information	332
A.13.1 Cardholder Account Information.....	333
A.13.2 Merchant Risk Indicator	338
A.13.3 3DS Requestor Authentication Information	340
A.13.4 3DS Requestor Prior Transaction Authentication Information	343
A.13.5 ACS Rendering Type	345
JSON Object Example	346
A.13.6 Device Rendering Options Supported	347
A.13.7 Challenge Data Entry	349
A.13.8 Transaction Status Conditions	351
A.13.9 Multi-Transaction	353
JSON Object Example	356
A.13.10 Seller Information.....	357
JSON Object Example	360
A.14 UI Data Elements	362
A.14.1 Issuer Image	366
A.14.2 Payment System Image	367
A.15 iframe and Sandbox Attributes	368
A.16 3-D Secure Array Fields	370
A.17 EMV Payment Token Information	371
A.18 Challenge Text Box Settings	372
A.19 Broadcast Information	375
A.20 Cardholder Information Text	377
A.21 SPC Transaction Data.....	379
A.22 HTTP Headers	384
HTTP Header Examples	384
Annex B Message Format.....	385
B.1 AReq Message Data Elements.....	385

B.2 ARes Message Data Elements.....	390
B.3 CReq Message Data Elements	392
B.4 CRes Message Data Elements.....	393
B.5 Final CRes Message Data Elements	394
B.6 PReq Message Data Elements.....	395
B.7 PRes Message Data Elements.....	396
B.8 RReq Message Data Elements	397
B.9 RRes Message Data Elements.....	398
B.10 OReq Message Data Elements	399
B.11 ORes Message Data Elements	400
B.12 Error Messages Data Elements	401
Annex C Generate ECC Key Pair.....	402
C.1 Input.....	402
C.2 Output	402
C.2.1 Process.....	402
Annex D Approved Transport Layer Security Versions	404
D.1 Cipher Suites for TLS 1.2	404
D.1.1 Supported Cipher Suites	404
D.1.2 Other Cipher Suites	404
D.2 Not Supported	404
Requirements	406

Figures

Figure 2.1: 3-D Secure Domains and Components	35
Figure 2.2: Frictionless Flow	43
Figure 2.3: 3DS Requestor Environment—Frictionless Flow—App-based	44
Figure 2.4: 3DS Requestor Environment—Browser-based.....	45
Figure 2.5: 3DS Requestor Environment—3RI.....	46
Figure 2.6: Challenge Flow	47
Figure 3.1: 3-D Secure Processing Flow Steps—App-based	50
Figure 3.2: Out-of-Band Processing Flow.....	65
Figure 3.3: 3-D Secure Processing Flow Steps—Browser-based.....	69
Figure 3.4: 3-D Secure Processing Flow Steps—3RI-based	82
Figure 3.5: 3-D Secure Processing Flow Steps—SPC-based	90
Figure 4.1: UI Template Zones—Portrait.....	95
Figure 4.2: UI Template Zones—Landscape	96
Figure 4.3: UI Template Examples—All Device Channels	97
Figure 4.4: UI Template Examples—App-based—Landscape.....	97
Figure 4.5: Sample UI OTP/Text Template with/without Device Header.....	98
Figure 4.6: 3DS SDK Options	99
Figure 4.7: Sample App-based Processing Screen—Portrait	100
Figure 4.8: Sample App-based Processing Screen—Landscape	100
Figure 4.9: Sample OTP/Text Template—App-based Processing Flow	101
Figure 4.10: Sample OOB Template (Manual transfer to and from the OOB App)—App-based Processing Flow.....	102
Figure 4.11: Sample OOB Template (OOB App and 3DS Requestor App on same device)—w/o OOB App launch button—App-based Processing Flow	102
Figure 4.12: Sample OOB Template (OOB App and 3DS Requestor App on same device with OOB App launch button)—App-based Processing Flow	103
Figure 4.13: Sample Decoupled Authentication Template—App-based Processing Flow	104
Figure 4.14: Sample Native UI OTP/Text Template—PA—Portrait	107
Figure 4.15: Sample Native UI OTP/Text Template—PA—Landscape	108
Figure 4.16: Sample Native UI with Optional Second OTP/Text entries Template—PA—Portrait.....	109
Figure 4.17: Sample Native UI with Optional Second OTP/Text entries Template—PA—Landscape.....	109
Figure 4.18: Sample Native UI OTP/Text Template—NPA.....	110
Figure 4.19: Sample Native UI—Single-select Information—PA—Portrait.....	111
Figure 4.20: Sample Native UI—Single-select Information—PA—Landscape	111
Figure 4.21: Sample Native UI—Multi-select Information—PA—Portrait	112
Figure 4.22: Sample Native UI—Multi-select Information—PA—Landscape	113
Figure 4.23: Sample OOB Native UI Template with Complete button—PA—Portrait.....	114
Figure 4.24: Sample OOB Native UI Template with Complete button—PA—Landscape.	114
Figure 4.25: Sample OOB Native UI Template with Automatic OOB App URL link—Portrait	115

Figure 4.26: Sample OOB Native UI Template with Automatic OOB App URL link—Landscape.....	115
Figure 4.27: Sample Challenge Information Text Indicator—PA.....	116
Figure 4.28: Sample Trust List/Device Binding Information Text—PA—Portrait	117
Figure 4.29: Sample Trust List/Device Binding Information Text—PA—Portrait	118
Figure 4.30: Sample Trust List/Device Binding Information Text—PA—Landscape	118
Figure 4.31: Sample Trust List/Device Binding Information Text—PA—Landscape	119
Figure 4.32: Sample Information Native UI Template—PA—Portrait.....	120
Figure 4.33: Sample Information Native UI Template—PA—Landscape.....	120
Figure 4.34: Sample Challenge Data Entry Masking—PA.....	121
Figure 4.35: Sample Data Entry Masking with Toggle	121
Figure 4.36: Native UI OTP/Text Template with Challenge Additional Label—PA—Portrait	122
Figure 4.37: Sample Native UI OTP/Text Template with Challenge Additional Label—PA—Landscape.....	123
Figure 4.38: Sample HTML UI OTP/Text Template—PA—Portrait.....	126
Figure 4.39: Sample HTML UI OTP/Text Template—PA—Landscape	126
Figure 4.40: Sample UI OTP/Text Template—NPA.....	127
Figure 4.41: Sample OOB HTML UI Template with Complete button—PA—Portrait	128
Figure 4.42: Sample OOB HTML UI Template with Complete button—PA—Landscape	129
Figure 4.43: Sample OOB HTML UI Template with OOB App URL button—PA—Portrait	130
Figure 4.44: Sample OOB HTML UI Template with OOB App URL button—PA—Landscape	131
Figure 4.45: Sample Information HTML UI Template—Portrait	132
Figure 4.46: Sample Information HTML UI Template—Landscape.....	133
Figure 4.47: Sample HTML Other UI Template—PA—Portrait.....	134
Figure 4.48: Sample HTML Other UI Template—PA—Landscape	135
Figure 4.49: App-based HTML and Browser UI Comparison.....	139
Figure 4.50: Sample Browser Lightbox Processing Screen without White Box.....	140
Figure 4.51: Sample Inline Browser Processing Screen with White Box.....	141
Figure 4.52: Sample Browser with Lightbox UI—PA.....	142
Figure 4.53: Sample Browser with Inline UI—PA	143
Figure 6.1: Security Flow—App-based	172
Figure 6.2: Security Flow—Browser-based	173
Figure 6.3: Security Functions.....	177
Figure A.1 Sample Cardholder Information UI Example—App—Portrait	377
Figure A.2 Sample Cardholder Information UI Example—Browser—Landscape	378

Tables

Table 1.1: Normative References.....	18
Table 1.2: ISO Standards.....	20
Table 1.3: Definitions	21
Table 1.4: Abbreviations	31
Table A.1: EMV 3-D Secure Data Elements.....	188
Table A.2: 3DS Method Data	308
Table A.3: 3DS CReq/CRes POST Data.....	310
Table A.4: Error Code, Error Description, and Error Detail	314
Table A.5: Excluded Currency Code and Country Code Values	319
Table A.6: Card Range Data	320
Table A.7: DS URLs.....	326
Table A.8: Supported Message Extension	327
Table A.9: Message Extension Attributes.....	331
Table A.10: Cardholder Account Information.....	333
Table A.11: Merchant Risk Indicator	338
Table A.12: 3DS Requestor Authentication Information	341
Table A.13: 3DS Requestor Prior Transaction Authentication Information.....	343
Table A.14: ACS Rendering Type	345
Table A.15: Device Rendering Options Supported	347
Table A.16: Challenge Data Entry	349
Table A.17: Transaction Status Conditions	351
Table A.18: Multi-Transaction	353
Table A.19: Seller Information.....	357
Table A.20: UI Data Elements.....	362
Table A.21: Issuer Image	366
Table A.22: Payment System Image	367
Table A.23: iframe Attributes.....	368
Table A.24: Sandbox Attributes.....	369
Table A.25: Token Information	371
Table A.26: Text Box Settings.....	373
Table A.27: Broadcast Information	375
Table A.28: SPC Transaction Data	379
Table A.29: HTTP Headers.....	384
Table B.1: AReq Data Elements	385
Table B.2: ARes Data Elements.....	390
Table B.3: CReq Data Elements	392
Table B.4: CRes Data Elements	393
Table B.5: Final CRes Data Elements	394
Table B.6: PReq Data Elements	395
Table B.7: PRes Data Elements.....	396
Table B.8: RReq Data Elements	397

Table B.9: RRes Data Elements	398
Table B.10: OReq Data Elements	399
Table B.11: ORes Data Elements	400
Table B.12: Error Message Data Elements	401

1 Introduction

The 3-D Secure authentication protocol is based on a three-domain model where the Acquirer Domain and Issuer Domain are connected by the Interoperability Domain for the purpose of authenticating a Cardholder during an electronic commerce (e-commerce) transaction or to provide identity verification and account confirmation.

The 3-D Secure authentication protocol supports two Message Categories:

- **Payment Authentication**—Cardholder authentication during an e-commerce transaction.
- **Non-Payment Authentication**—Identity verification and account confirmation.

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

- **App-based**—Authentication during a transaction on a Consumer Device that originates from an App provided by a 3DS Requestor (merchant, digital wallet, et al.). For example, an e-commerce transaction originating during a checkout process within a merchant's app.
- **Browser-based**—Authentication during a transaction on a Consumer Device that originates from a website utilising a Browser as defined in Table 1.3. For example, an e-commerce transaction originating during a checkout process within a website on a Consumer Device.
- **3DS Requestor Initiated**—Confirmation of account information and Cardholder authentication with no direct Cardholder present. For example, a subscription-based e-commerce merchant confirming that an account is still valid or Cardholder authentication when the 3DS Requester and the ACS utilises Decoupled Authentication.

1.1 Purpose

The purpose of this *EMV® 3-D Secure Protocol and Core Functions Specification* (hereinafter referred to as the *Core Specification*) is to describe the EMV 3-D Secure infrastructure and components, and to specify the requirements for each component within the infrastructure and their interaction.

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

1.2 Audience

This document is intended for stakeholders developing EMV 3-D Secure products and supporting 3-D Secure implementations.

1.3 Normative References

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

Table 1.1: Normative References

Reference	Publication Name	Bookmark
IETF BCP 47	<i>Tags for Identifying Languages</i>	https://tools.ietf.org/html/bcp47
ITU; ITU-T. E.164	<i>The International Public Telecommunication Numbering Plan</i>	https://www.itu.int/rec/T-REC-E.164/en
RFC 2045	<i>Multipurpose Internet Mail Extensions (MIME) Part One: Format of Internet Message Bodies</i>	https://tools.ietf.org/html/rfc2045
RFC 2397	<i>The "data" URL scheme</i>	https://datatracker.ietf.org/doc/html/rfc2397
RFC 2616	<i>Hypertext Transfer Protocol -- HTTP/1.1</i>	https://tools.ietf.org/html/rfc2616
RFC 3447	<i>PKCS #1: RSA Cryptography Specifications</i>	https://www.ietf.org/rfc/rfc3447.txt
RFC 3986	<i>Uniform Resource Identifier (URI): Generic Syntax</i>	https://tools.ietf.org/html/rfc3986
RFC 4122	<i>A Universally Unique IDentifier (UUID) URN Namespace</i>	https://tools.ietf.org/html/rfc4122
RFC 4158	<i>Internet X.509 Public Key Infrastructure: certification Path Building</i>	https://tools.ietf.org/html/rfc4158
RFC 5246	<i>The Transport Layer Security (TLS) Protocol Version 1.2</i>	https://tools.ietf.org/html/rfc5246
RFC 5322	<i>Internet Message Format</i>	https://tools.ietf.org/html/rfc5322
RFC 7159	<i>The JavaScript Object Notation (JSON) Data Interchange Format</i>	https://tools.ietf.org/html/rfc7159
RFC 7231	<i>Hypertext Transfer Protocol (HTTP/1.1): Semantics and Content</i>	https://tools.ietf.org/html/rfc7231
RFC 7233	<i>Hypertext Transfer Protocol (HTTP/1.1): Range Requests</i>	https://datatracker.ietf.org/doc/html/rfc7233
RFC 7515	<i>JSON Web Signatures (JWS)</i>	https://tools.ietf.org/html/rfc7515

Reference	Publication Name	Bookmark
RFC 7516	<i>JSON Web Encryption (JWE)</i>	https://tools.ietf.org/html/rfc7516
RFC 7517	<i>JSON Web Key (JWK)</i>	https://tools.ietf.org/html/rfc7517
RFC 7518	<i>JSON Web Algorithms (JWA)</i>	https://tools.ietf.org/html/rfc7518
RFC 8446	<i>The Transport Layer Security (TLS) Protocol Version 1.3</i>	https://tools.ietf.org/html/rfc8446
RFC 791	<i>INTERNET PROTOCOL</i>	https://tools.ietf.org/html/rfc791
RFC 4291	<i>IP Version 6 Addressing Architecture</i>	https://tools.ietf.org/html/rfc4291

1.4 Acknowledgement

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

Table 1.2: ISO Standards

Reference	Publication Name	Bookmark
ISO 3166	<i>Country Codes</i> —ISO 3166	http://www.iso.org/iso/country_codes
ISO 4217	<i>Currency Codes</i> —ISO 4217	http://www.iso.org/iso/home/standards/currency_codes.htm
ISO/IEC 7812-1	<i>ISO/IEC 7812-1 Identification cards—Identification of issuers—Part 1: Numbering system</i>	http://www.iso.org/iso/home/store/catalogue_tc/catalogue_detail.htm?csnumber=66011
ISO/IEC 7813	<i>ISO/IEC 7813 Information technology—Identification cards—Financial transaction cards</i>	http://www.iso.org/iso/home/store/catalogue_tc/catalogue_detail.htm?csnumber=43317
ISO/IEC 7816-5	<i>ISO/IEC 7816-5 Identification cards—Integrated circuit cards—Part 5: Registration of application providers</i>	https://www.iso.org/standard/34259.html
ISO 8583-1	<i>ISO 8583-1 Financial transaction card originated messages — Interchange message specifications — Part 1: Messages, data elements and code values</i>	https://www.iso.org/standard/31628.html
ISO/IEC 15946-1	<i>ISO/IEC 15946-1 Information technology—Security techniques—Cryptographic techniques based on elliptic curves—Part 1: General</i>	http://www.iso.org/iso/home/store/catalogue_tc/catalogue_detail.htm?csnumber=65480

1.5 Definitions

The following terms are used in this specification:

Table 1.3: Definitions

Term	Definition
3DS Client	The consumer-facing component allowing consumer interaction with the 3DS Requestor for initiation of the EMV 3-D Secure protocol.
3DS Integrator	An EMV 3-D Secure participant that facilitates and integrates the 3DS Requestor Environment, and optionally facilitates integration between the Merchant and the Acquirer.
3DS Method	A scripting call provided by the 3DS Integrator that is placed on the 3DS Requestor website. Optionally used to obtain additional Browser information to facilitate risk-based decisioning.
3DS Requestor	The initiator of the EMV 3-D Secure Authentication Request. For example, this may be a merchant or a digital wallet requesting authentication within a purchase flow.
3DS Requestor App	An App on a Consumer Device that can process a 3-D Secure transaction through the use of a 3DS SDK. The 3DS Requestor App is enabled through integration with the 3DS SDK.
3DS Requestor Environment	The 3DS Requestor-controlled components (3DS Requestor App, 3DS SDK, and 3DS Server) are typically facilitated by the 3DS Integrator. Implementation of the 3DS Requestor Environment will vary as defined by the 3DS Integrator.
3DS Requestor Initiated (3RI)	3-D Secure transaction initiated by the 3DS Requestor for the purposes of confirming that an account is still valid or for Cardholder authentication. The first main use case being recurring transactions (TV subscriptions, utility bill payments, etc.) where the merchant wants to perform a payment transaction to receive authentication data for each bill or a non-payment transaction to verify that a subscription user still has a valid form of payment. The second main use case is when the 3DS Requestor requests Decoupled Authentication as a method to authenticate the Cardholder.
3DS Requestor Website	Component that provides the website that requests Cardholder credentials (whether on file or entered by Cardholder).
3DS SDK	A component that interacts with the 3DS Requestor App. The 3DS SDK performs functions related to 3-D Secure on behalf of the 3DS Server.
3DS Server	Refers to the 3DS Integrator's server or systems that handle online transactions and facilitates communication between the 3DS Requestor and the DS.
3-D Secure (3DS)	An e-commerce authentication protocol that enables the secure processing of payment, non-payment and account confirmation card transactions.

Term	Definition
Abandon	The act of a Cardholder leaving a transaction by use of the Cancel action while in the process of a challenge. For example, using the Cancel button in the App challenge UI.
Absent	<p>Used in this specification to indicate that an element is absent when the name/value pair does not occur in the message.</p> <p>For example, element "firstName" is absent in the following JSON instance:</p> <pre data-bbox="514 586 779 698"> { "lastName": "Smith" } </pre>
Access Control Server (ACS)	A component that operates in the Issuer Domain, that verifies whether authentication is available for a card number and device type and authenticates specific Cardholders.
Access Control Server User Interface (ACS UI)	The ACS UI is generated during a Cardholder challenge and is rendered by the ACS within a Browser challenge iframe.
Acquirer	A financial institution that establishes a contractual service relationship with a Merchant for the purpose of accepting payment cards. In the context of 3-D Secure, in addition to the traditional role of receiving and sending authorisation and settlement messages (enters transaction into interchange), the Acquirer also determines whether a Merchant is eligible to support the Merchant's participation in 3-D Secure.
Acquirer Domain	Contains the systems and functions of the 3DS Requestor Environment and, optionally the Acquirer.
App Screen Orientation	<p>The orientation of the app screen display on the device, which may differ from the device orientation (for example, if the app supports Portrait-only or Landscape-only display, or if the device is in multi-window or split-screen mode).</p> <p>The orientation is considered Landscape if the display is wider than it is tall, and Portrait otherwise.</p>
Attempts	In this specification, used to indicate the process by which proof of an authentication attempt is generated when payment authentication is not available. Support for Attempts is determined by each DS.
Authentication	In the context of 3-D Secure, the process of confirming that the person making an e-commerce transaction is entitled to use the payment card.
Authentication Request (AReq) Message	An EMV 3-D Secure message sent by the 3DS Server via the DS to the ACS to initiate the authentication process.
Authentication Response (ARes) Message	An EMV 3-D Secure message returned by the ACS via the DS in response to an Authentication Request message.

Term	Definition
Authentication Value (AV)	A cryptographic value generated by the ACS to provide a way, during authorisation processing, for the authorisation system to validate the integrity of the authentication result. The AV algorithm is defined by each Payment System.
Authorisation	A process by which an Issuer, or a processor on the Issuer's behalf, approves a transaction for payment.
Authorisation System	The systems and services through which a Payment System delivers online financial processing, authorisation, clearing, and settlement services to Issuers and Acquirers.
Bank Identification Number (BIN)	The first six or eight digits of a payment card account number that uniquely identifies the issuing financial institution. Also referred to as Issuer Identification Number (IIN) in ISO 7812.
Base64	Encoding applied to the Authentication Value data element as defined in RFC 2045.
Base64url	Encoding applied to the 3DS Method Data, Device Information, WebAuthn Credential List and the CReq/CRes messages as defined in RFC 7515.
Browser	A Browser is a dedicated software application for accessing information on the World Wide Web, for example Chrome, Safari, Edge, Firefox. When a user requests a web page from a particular website, the Browser retrieves the necessary content from a web server and then displays the page on the consumer's screen. In the context of 3-D Secure, the Browser is a conduit to transport messages between the Acquirer Domain and the Issuer Domain. A Browser is distinguished from a UI component for example, a WebView, or Custom Tabs, which can be used to display content within an App on a mobile device. The Browser flow is invoked by a Browser whereas the EMVCo specification does not support a UI component within an app invoking the Browser flow.
Card	In this specification, synonymous to the account of a payment card.
Card Range Data File	The file containing the JSON Card Range Data object. The Card Range Data provides to the 3DS Server the 3DS protocol versions supported by the card ranges hosted by the ACS, and other optional information (e.g. 3DS Method, Message Extension).
Cardholder	An individual to whom a card is issued or who is authorised to use that card.
Certificate	An electronic document that contains the public key of the certificate holder and which is attested to by a Certificate Authority (CA) and rendered not forgeable by cryptographic technology (signing with the private key of the CA).
Certificate Authority (CA)	A trusted party that issues and revokes certificates. Refer also to DS Certificate Authority.
Challenge	The process where the ACS is in communication with the 3DS Client to obtain additional information through Cardholder interaction.

Term	Definition
Challenge Flow	A 3-D Secure flow that involves Cardholder interaction as defined in Section 2.5.2.
Challenge Request (CReq) Message	An EMV 3-D Secure message sent by the 3DS SDK or 3DS Server where additional information is sent from the Cardholder to the ACS to support the authentication process.
Challenge Response (CRes)	The ACS response to the CReq message. It can indicate the result of the Cardholder authentication or, in the case of an App-based model, also signal that further Cardholder interaction is required to complete the authentication.
Consumer Device	Device used by a Cardholder such as a smartphone, laptop, or tablet that the Cardholder uses to conduct payment activities including authentication and purchase.
Decoupled Authentication	<p>Decoupled Authentication is an authentication method whereby authentication can occur independent from the Cardholder's experience with the 3DS Requestor. The authentication method used for Decoupled Authentication is outside the scope of this specification. However, one method could be a push notification to a banking app that completes authentication and then sends the results to the ACS.</p> <p>Decoupled Authentication is applicable to all Device Channels.</p>
Decoupled Authentication Fallback	An additional challenge option for an ACS during the Challenge process. By returning Transaction Status = D in the RReq message, the ACS requests that the 3DS Server initiate a subsequent 3DS authentication with Decoupled Authentication supported.
Device Binding	In this specification, the process to link the Consumer Device used for a transaction to the Cardholder Account and/or Cardholder.
Device Channel	<p>Indicates the channel from which the transaction originated. Either:</p> <ul style="list-style-type: none"> • App-based (01-APP) • Browser-based (02-BRW) • 3DS Requestor Initiated (03-3RI)
Device Information	Data provided by the Consumer Device that is used in the authentication process.
Digital signature	An asymmetric cryptographic method whereby the recipient of the data can prove the origin and integrity of data, thereby protecting the sender of the data and the recipient against modification or forgery by third parties and the sender against forgery by the recipient.
Digital wallet	A software component that allows a user to make an electronic payment with a financial instrument (such as a credit card) while hiding the low-level details of executing the payment protocol, including such tasks as entering an account number and providing shipping information and Cardholder identifying information.

Term	Definition
Directory Server (DS)	A server component operated in the Interoperability Domain; it performs a number of functions that include: authenticating the 3DS Server, routing messages between the 3DS Server and the ACS, and validating the 3DS Server, the 3DS SDK, and the 3DS Requestor.
Directory Server Certificate Authority (DS CA)	A component that operates in the Interoperability Domain; generates and distributes selected digital certificates to components participating in 3-D Secure. Typically, the Payment System to which the DS is connected operates the CA.
Directory Server ID (directoryServerID)	<p>Registered Application Provider Identifier (RID) that is unique to the Payment System. RIDs are defined by the ISO 7816-5 standard.</p> <p>The Directory Server ID is a hex value encoded as a 10-character text. For example, 0x'A000000003' is encoded as 'A000000003'.</p>
Electronic Commerce Indicator (ECI)	Payment System-specific value provided by the ACS to indicate the results of the attempt to authenticate the Cardholder.
Empty	<p>An element is empty if the field name is present and the value is empty. For example, element "firstName" has no data in the following JSON instance.</p> <pre data-bbox="514 983 1006 1118">{ "firstName": "", "lastName": "Smith" }</pre>
EMV	A term referring to EMVCo's specifications for global interoperability and acceptance of secure payment transactions and/or products and services complying with such specifications.
EMV Payment Token	As defined in the EMV Tokenisation Specification, a surrogate value for a PAN that is a variable length, ISO/IEC 7812-compliant numeric issued from a designated Token BIN or Token BIN Range and flagged accordingly in all appropriate BIN tables. A Payment Token passes basic validation rules of an account number, including the Luhn check digit. Payment Tokens do not have the exact same value as or conflict with a PAN.
EMVCo	EMVCo, LLC, a limited liability company incorporated in Delaware, USA.
Ends 3-D Secure Processing	<p>In the 3-D Secure processing flow, this indicates that no further processing as defined by this specification will be performed.</p> <p>Per merchant preferences, an authorisation transaction may still be performed although it will happen without a successful 3-D Secure authentication outcome.</p>

Term	Definition
Ends processing	<p>In the 3-D Secure processing flow, this indicates that an error has been found by a specific 3-D Secure component, which reports the error via the appropriate Error Message as defined in Section A.5.5 or RReq message as defined in Table B.8.</p> <p>The specific 3-D Secure component reports the error to the component from which the erroneous message was received, and may inform other components about the error and will stop further 3-D Secure processing.</p> <p>The subsequent 3-D Secure components in the authentication flow will still perform further execution of the received message with an Error message to close the error situation. For an RReq message, the sending component should expect back an RRes message before stopping 3-D secure processing.</p>
FIDO Authenticator	<p>An authentication entity that meets the FIDO Alliance's requirements and which has related metadata. A FIDO Authenticator is responsible for user verification, and maintaining the cryptographic material required for the relying party authentication. For additional information, refer to: https://fidoalliance.org.</p>
Frictionless	<p>The process of authentication achieved without Cardholder interaction.</p>
Frictionless Flow	<p>A 3-D Secure flow that does not involve Cardholder interaction as defined in Section 2.5.1.</p>
Fully Qualified URL	<p>A Fully Qualified URL contains all the information necessary to locate a web resource, and is defined as an 'Absolute-URL string' with scheme 'https', encoded in 'UTF-8' using 'url-code-points' from https://whatwg.org/</p> <p>Refer to https://url.spec.whatwg.org/#absolute-url-string and to https://url.spec.whatwg.org/#url-code-points</p> <p>A Fully Qualified URL does not contain credentials (https://url.spec.whatwg.org/#include-credentials)</p> <p>Example: https://server.domainname.com/acs/auth%20(*ret)</p>
iframe	<p>An iframe (short for inline frame) is a frame within a frame. It is used to embed a piece of HTML content from other sources in an HTML document.</p> <p>Refer to:</p> <p>w3c: https://www.w3.org/html/wg/spec/the-iframe-element.html#the-iframe-element OR</p> <p>whatwg: https://html.spec.whatwg.org/#the-iframe-element</p>
Information Only	<p>Information Only is a Transaction Status value, whereby the ACS acknowledges the 3DS Requestor's preference to not challenge on the transaction since the data sent was only for informational purposes.</p>

Term	Definition
Interaction Counter	The number of interactions for each transaction is tracked by the ACS and sent with the RReq message to the Directory Server (DS). Used by the ACS to set a maximum number of Cardholder interactions as determined by the selected Challenge Flows and security requirements to allow an appropriate number of Cardholder retries without going beyond a pre-set maximum.
Interoperability Domain	Facilitates the transfer of information between the Issuer Domain and Acquirer Domain systems.
Issuer	A financial institution that issues payment cards, contracts with Cardholders to provide card services, determines eligibility of Cardholders to participate in 3-D Secure, and identifies for the Directory Server card number ranges eligible to participate in 3-D Secure.
Issuer Domain	Contains the systems and functions of the Issuer and its customers (Cardholders).
JavaScript Object Notation (JSON)	An open standard format that uses human-readable text to transmit data objects consisting of attribute-value pairs. It is typically used to transmit data between a server and web application. Refer to Table 1.1 for RFC references.
Key	In cryptography, the value needed to encrypt and/or decrypt a value.
Key management	The handling of cryptographic keys and other security parameters during the entire lifetime of the keys, including generation, storage, entry and use, deletion or destruction, and archiving.
MAC	Message Authentication Code. A symmetric (secret key) cryptographic method that protects the sender and recipient against modification and forgery of data by third parties.
Merchant	Entity that contracts with an Acquirer to accept payment cards. Manages the online shopping experience with the Cardholder, obtains card number, and then transfers control to the 3DS Server, which conducts payment authentication.
Message Category	Indicates the category of the EMV 3-D Secure message. Either: <ul style="list-style-type: none">• Payment (01-PA), or• Non-Payment (02-NPA)
Message Version	Refers to the protocol version that will be used by all components to process the 3-D Secure transaction. Message version is always consistent across all 3-D Secure protocol messages for a specific transaction.

Term	Definition
Missing	<p>An element is missing either if it is absent (that is the name/value pair does not occur in the message) or if the field name is present and value is empty. For example, element "firstName" has no data in both of the following JSON instances.</p> <p>Example of empty field name present and value empty:</p> <pre data-bbox="514 489 1006 646">{ "firstName": "", "lastName": "Smith" }</pre> <p>Example of absent name/value pair:</p> <pre data-bbox="514 676 784 788">{ "lastName": "Smith" }</pre>
Name/value Pair (NVP)	A simple class encapsulating an attribute/value pair.
Native	Refers to the original method for a device display, utilising its own APIs.
Null	<p>An element is null if the field name is present and the value is null. For example, element "firstName" has null in the following JSON instance.</p> <pre data-bbox="514 1035 1054 1147">{ "firstName": null, "lastName": "Smith" }</pre>
One-Time Passcode (OTP)	A passcode that is valid for only one login session or transaction, on a computer system or other digital device.
Out-of-Band (OOB)	A Challenge activity that is completed outside of, but in parallel to, the 3-D Secure flow. The final Challenge Request is not used to carry the data to be checked by the ACS but signals only that the authentication has been completed. ACS authentication methods or implementations are not defined by the 3-D Secure specification.
OOB Authentication App	App on a Consumer Device that is used by the ACS to authenticate the Cardholder as part of the 3-D Secure flow, for example, a mobile banking app. See Section 3.2 for details of the OOB flow.
Operation Request (OReq) Message	The OReq message sequence is created to communicate operational information serving as an alert, a reminder, report, or call to action. This message is not part of the 3-D Secure authentication message flow.
Operation Response (ORes) Message	The ORes message acknowledges receipt of the OReq message sequence. The message is created by the recipient of the OReq message and sent to the source of the OReq message.
Payment System	A Payment System defines the operating rules and conditions, and the requirements for card issuance and Merchant acceptance.

Term	Definition
Platform Provider	An entity that provides a digital ecosystem consisting of an operating system and/or hardware components, capable of uniquely identifying the consumer and their device through a user ID and a hardware-derived device ID, and sharing these IDs for the purposes of risk assessment and fraud prevention.
Preparation Request (PReq) Message	3-D Secure message sent from the 3DS Server to the DS to request the ACS and DS Protocol Version(s) that correspond to the DS card ranges as well as an optional 3DS Method URL to update the 3DS Server's internal storage information.
Preparation Response (PRes) Message	Response to the PReq message that contains the DS Card Ranges, active Protocol Versions for the ACS and DS and 3DS Method URL, or a Card Range Data File URL to download this information, so that updates can be made to the 3DS Server's internal storage.
Private key	Part of an asymmetric cryptographic system. The key that is kept secret and known only to an owner.
Proof of authentication attempt	Refer to Attempts.
Protocol Version	Defines the message interoperability between the EMV 3-D Secure components.
Public key	Part of an asymmetric cryptographic system. The key known to all parties.
Public key pair	Two mathematically related keys—a public key and a private key—that are used with a public key (asymmetric) cryptographic algorithm to permit the secure exchange of information without the necessity for a secure exchange of a secret.
Registered Application Provider Identifier (RID)	<p>Registered Application Provider Identifier (RID) is unique to a Payment System.</p> <p>RIDs are defined by the ISO 7816-5 standard and are issued by the ISO/IEC 7816-5 registration authority.</p> <p>RIDs are 5 bytes.</p>
Responsive Design	<p>Responsive design is an approach to make the web page content adjust to the dimensions of the device's screen for a better user experience.</p> <p>The approach is based on the use of three web techniques when designing the web pages:</p> <ul style="list-style-type: none"> • Flexible grid to create the web page layout that dynamically adapt to the screen width. • Media queries to allow the page to adopt different CSS styles depending on the Browser and device screen. • Flexible media to make images scalable to the size of the viewport.
Results Request (RReq) Message	Message sent by the ACS via the DS to transmit the results of the authentication transaction to the 3DS Server.

Term	Definition
Results Response (RRes) Message	Message sent by the 3DS Server to the ACS via the DS to acknowledge receipt of the Results Request message.
Secret key	A key used in a symmetric cryptographic algorithm such as DES which, if disclosed publicly, would compromise the security of the system.
Secure Payment Confirmation	FIDO-based authentication to securely confirm payments initiated via the Payment Request API on a Browser (refer to w3.org for additional information).
Token Service Provider	A role within the Payment Tokenisation ecosystem that is authorised by a Token Programme to provide Payment Tokens to registered Token Requestors. Refer to the <i>EMV® Payment Tokenisation Specification - Technical Framework</i> .
Transport Layer Security (TLS)	A cryptographic protocol developed by the IETF (Internet Engineering Task Force) to confidentially transmit information over open networks, such as the Internet. Refer to Table 1.1 for RFC references.
Trust List	In this specification, the process of enabling the Cardholder to place the 3DS Requestor on their trusted beneficiaries list.
Uniform Resource Locator (URL)	Address scheme for pages on the World Wide Web usually in the format http://www.example.com or https://www.example.com .
Universal App Link	<p>Standard HTTPS links for opening a specific mobile app, installed on a device. The implementation is platform-specific.</p> <ul style="list-style-type: none"> • Android App Links: https://developer.android.com/training/app-links • iOS Universal Links: https://developer.apple.com/ios/universal-links
Universally Unique Identifier (UUID)	Identifier standard used in software construction. In its canonical form, a UUID is represented by 32 lowercase hexadecimal digits, displayed in five groups separated by hyphens, in the form 8-4-4-4-12 for a total of 36 characters (32 alphanumeric characters and four hyphens). Refer to Table 1.1 for RFC references.
Validate	In this specification, the process of checking a message against the requirements for presence and format of each data element in the message as defined in Table A.1 and detailed outline in Section 5.1.6. Refer to Section 5.9 for additional information.
Verify	In this specification, the process of checking a message cryptographically as defined in Section 6.2. Refer to Section 5.9 for additional information.
Wallet	Refer to Digital wallet.
WebAuthn	Defines an API enabling the creation and use of strong, attested, scoped, public key-based credentials by web applications, for the purpose of strongly authenticating users. Refer to https://www.w3.org/TR/webauthn-2/
X.509	Certificate format as defined in RFC 4158.

1.6 Abbreviations

The abbreviations listed in Table 1.4 are used in this specification.

Table 1.4: Abbreviations

Abbreviation	Description
3DS	Three Domain Secure
3DS SDK	Three Domain Secure Software Development Kit
3RI	3DS Requestor Initiated
ACS	Access Control Server
AOC	Attestation of Compliance
AReq	Authentication Request
ARes	Authentication Response
AV	Authentication Value
AVS	Address Verification Service
BIN	Bank Identification Number
CA	Certificate Authority
CEK	Content Encryption Key
CReq	Challenge Request
CRes	Challenge Response
DH	Diffie–Hellman
DS	Directory Server
DS CA	Directory Server Certificate Authority
ECC	Elliptic Curve Cryptography
ECI	Electronic Commerce Indicator
JSON	JavaScript Object Notation
LOA	Letter of Approval
MAC	Message Authentication Code
NPA	Non-Payment Authentication

Abbreviation	Description
NVP	Name/value pair
OOB	Out-of-Band
PA	Payment Authentication
OTP	One-time Passcode
OReq	Operation Request Message
ORes	Operation Response Message
PReq	Preparation Request Message
PRes	Preparation Response Message
RID	Registered Application Provider Identifier
RReq	Results Request Message
RRes	Results Response Message
RSA	Rivest–Shamir–Adleman
SDK	Software Development Kit
SPC	Secure Payment Confirmation
TLS	Transport Layer Security
URL	Uniform Resource Locator
UUID	Universally Unique Identifier

1.7 3-D Secure Protocol Version Number

Refer to *EMV® Specification Bulletin 255* for the list of active Protocol Version Numbers.

1.8 Supporting Documentation

The following documents are specific to the EMV 3-D Secure protocol. These documents as well as the *EMV® 3-D Secure Frequently Asked Questions* are located on the EMVCo website under the 3-D Secure heading.

- *EMV® 3-D Secure—SDK Specification*
- *EMV® 3-D Secure—Split-SDK Specification*
- *EMV® 3-D Secure SDK Technical Guide*

- *EMV® 3-D Secure SDK—Device Information*
- *EMV® 3-D Secure Message Extensions*
 - *EMV® 3-D Secure Bridging Message Extension*
 - *EMV® 3-D Secure Device Acknowledgement Message Extension*
 - *EMV® 3-D Secure Payment Token Message Extension*
 - *EMV® 3-D Secure Travel Industry Message Extension*
- *EMV® 3-D Secure JSON Message Samples*
- *EMV® 3-D Secure App-based Cryptographic Worked Samples*
- *EMV® 3-D Secure Browser Flow Best Practices*
- *EMV® Specification Bulletin 255—3-D Secure Protocol Version Numbers*

1.9 Terminology and Conventions

The following words are used often in this specification and have a specific meaning:

Shall

Defines a product or system capability which is mandatory.

May

Defines a product or system capability which is optional or a statement which is informative only and is out of scope for this specification.

Should

Defines a product or system capability which is recommended.

End(s) 3-D Secure Processing

As outlined in Chapter 3, defines a specific exception scenario in the 3-D Secure authentication flows where further processing is outside the scope of this specification. Refer to Table 1.3 for additional information.

End(s) Processing

As outlined in Chapter 3, defines a specific exception scenario in the 3-D Secure authentication flows where a 3-D Secure component experiences an error and does not process the transaction normally. Therefore, subsequent components take action on the error instance. Refer to Table 1.3 for additional information.

Increment(s)

A 3DS component may be required to increment a counter in which case the increment is increasing the counter by one.

3DS SDK

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

1. **Default SDK**—Software component designed as an SDK that is integrated into a 3DS Requestor App. This SDK option is defined in the *EMV 3-D Secure—SDK Specification*, in which it is referred to as the 3DS SDK. In earlier versions of this *Core Specification*, this is referred to as the 3DS SDK.

2. **Split-SDK**—Client-server implementation of the 3DS SDK. Some functions of the Split-SDK entity can be performed by either a Split-SDK Client or a Split-SDK Server or, in some situations, both. The Split-SDK has multiple variants depending on the Consumer Device and the 3DS Requestor Environment. These variants include the Split-SDK/Native, Split-SDK/Shell, and Split-SDK/Browser, and each is defined in the *EMV 3-D Secure—Split-SDK Specification*.

Unless explicitly noted otherwise, the term 3DS SDK applies as identified above. Refer to the applicable 3DS SDK specification for detailed information regarding the SDK options.

Activate(s) the 3DS SDK

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

Perform(s) the Challenge

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

1.10 Constraints

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

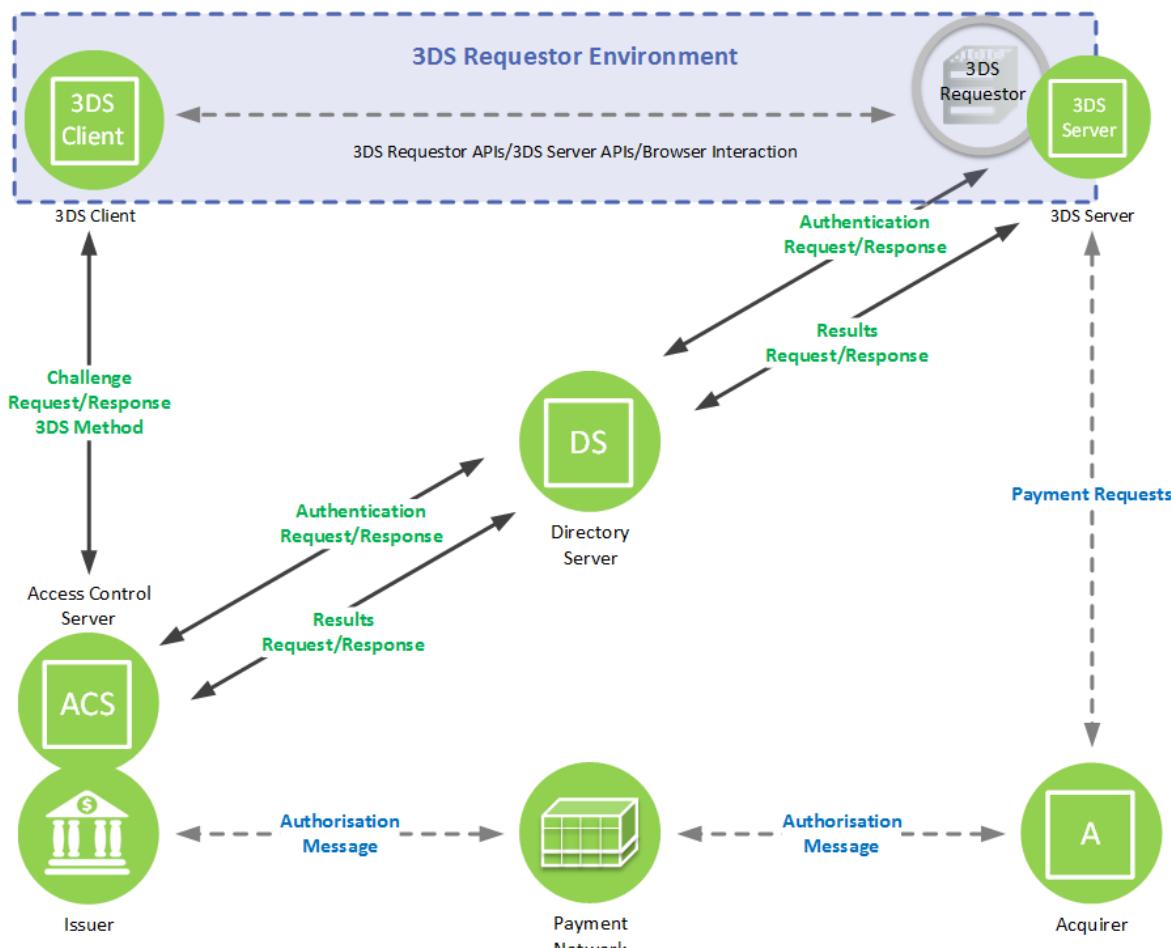
2 EMV 3-D Secure Overview

This overview describes the components, the systems, and the functions necessary to implement 3-D Secure. Descriptions are divided into the following domains:

- **Acquirer Domain**—3-D Secure transactions are initiated from the Acquirer Domain
- **Interoperability Domain**—3-D Secure transactions are switched between the Acquirer Domain and Issuer Domain
- **Issuer Domain**—3-D Secure transactions are authenticated in the Issuer Domain

Figure 2.1 depicts the interaction of the three domains and the components of each. Because the implementation of the 3DS Requestor Environment may vary, the diagram purposefully does not imply a specific implementation of these components or how they interoperate. For example, the 3DS Client may communicate directly with the 3DS Server, or the 3DS Server and 3DS Requestor may be functionally combined.

Figure 2.1: 3-D Secure Domains and Components



Note: Dashed arrows and 3DS Requestor are not part of 3DS specification and are shown for clarity only

2.1 Acquirer Domain

The Acquirer Domain has the following components:

- 3DS Requestor Environment
 - 3DS Requestor
 - 3DS Client
 - 3DS Server
- 3DS Integrator
- Acquirer (for Payment Authorisation)

2.1.1 3DS Requestor Environment

The 3DS Requestor Environment is a collective term for components under the 3DS Requestor's control that support 3-D Secure. The 3DS Requestor Environment components include:

- 3DS Requestor
- 3DS Client
- 3DS Server

2.1.1.1 3DS Requestor

The 3DS Requestor initiates the AReq message and is the conduit for the 3-D Secure data from the Consumer Device. For example, in payment authentication, the 3DS Requestor typically represents the existing Merchant web server for online shopping.

The 3DS Requestor has a relationship with the 3DS Client either via the 3DS Requestor App, or the 3DS Method/Browser on the Consumer Device. The 3DS Requestor has a link to or integration with the 3DS Server.

To process 3-D Secure transactions:

- **App-based**—3DS Requestor App integrates with the 3DS SDK as defined in the applicable 3DS SDK specification. The 3DS SDK displays the User Interface (UI) to Cardholders.
- **Browser-based**—3DS Requestor Browser-based solution utilises the 3DS Method to gather Browser information/device details and the ACS provides HTML to the Browser to display the UI to the Cardholder when a challenge is necessary.

2.1.1.2 3DS Client

The 3DS Client is the component on a Consumer Device that initiates a 3-D Secure authentication. For example, in payment authentication, the 3DS Client is integrated with the Merchant checkout as part of an online shopping experience.

The 3DS Client can be implemented in two models:

- **App-based**—The 3DS Client is the 3DS SDK that is integrated with the 3DS Requestor App and facilitates the Cardholder interaction.

The 3DS SDK gathers 3-D Secure information from the Consumer Device, supports the authentication of the Access Control Server (ACS), and protects the Cardholder authentication data flow.
- **Browser-based**—The 3DS Client is the 3DS Method that is integrated with the 3DS Requestor's website and is invoked within a Browser on the Consumer Device.

The 3DS Method is a scripting call provided by the 3DS Integrator placed on the website on which the Cardholder is interacting, such as a Merchant checkout page in a payment transaction. The purpose of the 3DS Method is to obtain additional Browser information to help facilitate risk-based decisioning.

2.1.1.3 3DS Server

The 3DS Server provides the functional interface between the 3DS Requestor Environment flows and the DS. The 3DS Server is responsible for:

- Collecting necessary data elements for 3-D Secure messages
- Authenticating the DS
- Validating the DS, the 3DS SDK, and the 3DS Requestor
- Ensuring that message contents are protected

To initiate a 3-D Secure authentication, the 3DS Server collects the necessary data elements from any or all of the components within the 3DS Requestor Environment. For example, in payment authentication, the Cardholder could provide account information using a Consumer Device, or the information could be held on file within the 3DS Requestor Environment. Device Information is obtained by the 3DS Client and forwarded to the 3DS Server.

Note: Following payment authentication, depending on the 3DS Requestor configuration, the 3DS Server may also link to the Acquirer and initiate authorisation requests.

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

The 3DS Integrator role provides the functional interface between the 3DS Requestor Environment and the 3-D Secure messages.

The role of the 3DS Integrator is to provision the 3DS Server and the 3DS Client, and to integrate the 3-D Secure functionality with the 3DS Requestor business functionality. This function is critical to interfacing with the DS and the ACS within the authentication messaging, while also acting as the conduit for challenge results when performed. The 3DS Integrator provides the approved 3DS SDK component or the 3DS Method functionality to 3DS Requestors for integration with their 3DS Requestor App and/or website.

The 3DS Integrator is responsible for registration of Merchants with all required DSSs.

Note: Following payment authentication, the 3DS Integrator can offer authorisation functionality with an Acquirer.

2.1.3 Acquirer (Payment Authorisation)

An Acquirer is a financial institution that:

- Enters into a contractual relationship with a Merchant for the purpose of accepting payment card transactions
- Supports the Merchant's participation in 3-D Secure

Following a 3-D Secure payment authentication, the Acquirer performs its traditional role, which involves:

- Receiving authorisation requests from the Merchant
- Sending authorisation requests to the authorisation system
- Providing authorisation responses to the Merchant
- Submitting the completed transactions to the settlement system

2.2 Interoperability Domain

The Interoperability Domain has the following components:

- Directory Server (DS)
- Directory Server Certificate Authority (DS CA)
- Authorisation System

2.2.1 Directory Server

The DS performs a number of functions that include:

- Authenticating the 3DS Server and the ACS
- Routing messages between the 3DS Server and the ACS
- Validating the 3DS Server, the 3DS SDK, and the 3DS Requestor
- Defining specific programme rules (for example, logos, time-out values, etc.)
- Onboarding 3DS Servers and ACSs
- Maintaining ACS and DS Protocol Version lists and 3DS Method URLs

2.2.2 Directory Server Certificate Authority

The DS CA generates the DS Public Key to the 3DS SDK and generates Transport Layer Security (TLS) certificates for use by 3-D Secure components. The DS CA is typically operated by the Payment System responsible for a specific DS.

These certificates include:

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

Refer to Chapter 6 for detailed information about certificates.

2.2.3 Authorisation System (Payment Authentication)

The Authorisation System performs its traditional role after payment authentication, which involves:

- Receiving authorisation requests from the Acquirer
- Sending authorisation requests to the Issuer
- Providing authorisation responses to the Acquirer
- Providing clearing and settlement services to the Acquirer and the Issuer

2.3 Issuer Domain

The Issuer Domain has the following components:

- Cardholder
- Consumer Device
- Issuer
- Access Control Server (ACS)

2.3.1 Cardholder

The Cardholder provides account information using a Consumer Device. If necessary, the Cardholder is prompted to provide additional information for authentication.

2.3.2 Consumer Device

The Consumer Device has the capability to run a 3DS Requestor App or present a website on a Browser that can be used for 3-D Secure authentication.

The Consumer Device-based components of the 3DS Requestor Environment depend on the model:

- App-based—the 3DS SDK integrated with the 3DS Requestor App
- Browser-based—a Browser utilising the 3DS Method

These components have a specific relationship with the 3DS Requestor and 3DS Server.

2.3.3 Issuer

An Issuer is a financial institution that:

- Enters into a contractual relationship with the Cardholder for issuance of one or more payment cards
- Defines card number ranges eligible to participate in 3-D Secure
- Provides card number ranges to be added to the applicable DS

2.3.4 Access Control Server

The ACS contains the authentication rules and is controlled by the Issuer. ACS functions include:

- Verifying whether a card number is eligible for 3-D Secure authentication
- Verifying whether a Consumer Device type is eligible for 3-D Secure authentication
- Authenticating the Cardholder or confirming account information

While these functions may belong to a single logical ACS, implementations may divide the processing by function or other characteristics (for example, card number range) among multiple physical servers.

2.4 3-D Secure Messages

This section introduces the messages defined for 3-D Secure. Refer to Chapter 5 for detailed information about message handling, and Annex B for message data elements.

2.4.1 Authentication Request Message (AReq)

The AReq message is the initial message in the 3-D Secure authentication flow. The 3DS Server forms the AReq message when requesting authentication of the Cardholder. It can contain Cardholder, payment, and Device information for the transaction. There is only one AReq message per authentication, except for the 3DS Requestor-Initiated SPC Authentication and Decoupled Authentication Fallback.

2.4.2 Authentication Response Message (ARes)

The ARes message is the Issuer's ACS response to the AReq message. It can indicate that the Cardholder has been authenticated, or that further Cardholder interaction is required to complete the authentication. There is only one ARes message per authentication, except for the 3DS Requestor-Initiated SPC Authentication and Decoupled Authentication Fallback.

2.4.3 Challenge Request Message (CReq)

The CReq message initiates Cardholder interaction in a Challenge Flow and can be used to carry authentication data from the Cardholder.

- **App-based**—The CReq message is sent by the 3DS SDK. There are two or more CReq messages per challenge as multiple back-and-forth attempts between the ACS and the Cardholder may be required to complete the authentication.
- **Browser-based**—The CReq message is formed by the 3DS Server and is posted through the Cardholder Browser. There is only one CReq message per challenge.

2.4.4 Challenge Response Message (CRes)

The CRes message is the ACS response to the CReq message. It can indicate the result of the Cardholder authentication or, in the case of an App-based model, also signal that further Cardholder interaction is required to complete the authentication.

- **App-based**—Elements of the CRes message provide the necessary data for the 3DS SDK to generate and display the user interface (UI) for the challenge. There are two or more CRes messages per transaction to complete Cardholder authentication.
- **Browser-based**—The CRes message contains the authentication result and completes the Cardholder challenge. There is only one CRes message per challenge.

2.4.5 Results Request Message (RReq)

The RReq message communicates the results of the authentication or verification. The message is sent by the ACS through the DS to the 3DS Server. There is only one RReq message per 3DS transaction. The RReq message is not present in a Frictionless transaction.

2.4.6 Results Response Message (RRes)

The RRes message acknowledges receipt of the RReq message. The message is sent by the 3DS Server through the DS to the ACS. There is only one RRes message per RReq message.

2.4.7 Preparation Request Message (PReq)

The PReq message is sent from the 3DS Server to the DS to request information about the ACSs and the DS. This message is not part of the 3-D Secure authentication message flow.

2.4.8 Preparation Response Message (PRes)

The PRes message is the DS response to the PReq message. The 3DS Server can utilise the PRes message to cache information about the ACSs and the DS (for example, about which Protocol Version(s) are supported). This message is not part of the 3-D Secure authentication message flow.

2.4.9 Operation Request Message (OReq)

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

2.4.10 Operation Response Message (ORes)

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

2.4.11 Error Message

Error messages provide additional information about an error that occurred during message processing between the 3DS Server, the DS, the ACS, and the 3DS SDK. Chapter 5, Annex A, and Annex B provide additional information about Error messages.

2.5 Authentication Flows

This section introduces the authentication flows defined for EMV 3-D Secure. Refer to Chapter 3 of this specification for authentication flow requirements.

2.5.1 Frictionless Flow

The Frictionless Flow initiates a 3-D Secure authentication flow and consists of an AReq message and an ARes message.

The Frictionless Flow does not require further Cardholder interaction to achieve a successful authentication and complete the 3-D Secure authentication process.

2.5.2 Challenge Flow

In addition to the AReq and ARes messages that comprise the Frictionless flow, the Challenge Flow completes with the RReq and RRes messages. The Challenge Flow also includes CReq and CRes messages except in the case of Decoupled Authentication. If the ACS determines that further Cardholder interaction is required to complete the authentication, the Frictionless Flow transitions into the Challenge Flow. For example, a challenge may be necessary because the transaction is deemed high-risk, is above certain thresholds, or requires a higher level of authentication due to country mandates (or regulations).

3DS Requestors decide whether to proceed with the challenge, or to terminate the 3-D Secure authentication process.

2.5.3 Processing EMV Payment Tokens

General configuration:

In order for this specification to appropriately handle flows that initiate with EMV Payment Tokens, the Payment Token ranges shall be shared and configured on the appropriate DS. This is essential for the EMV Payment Token transaction to be routed to the appropriate DS and then to the ACS.

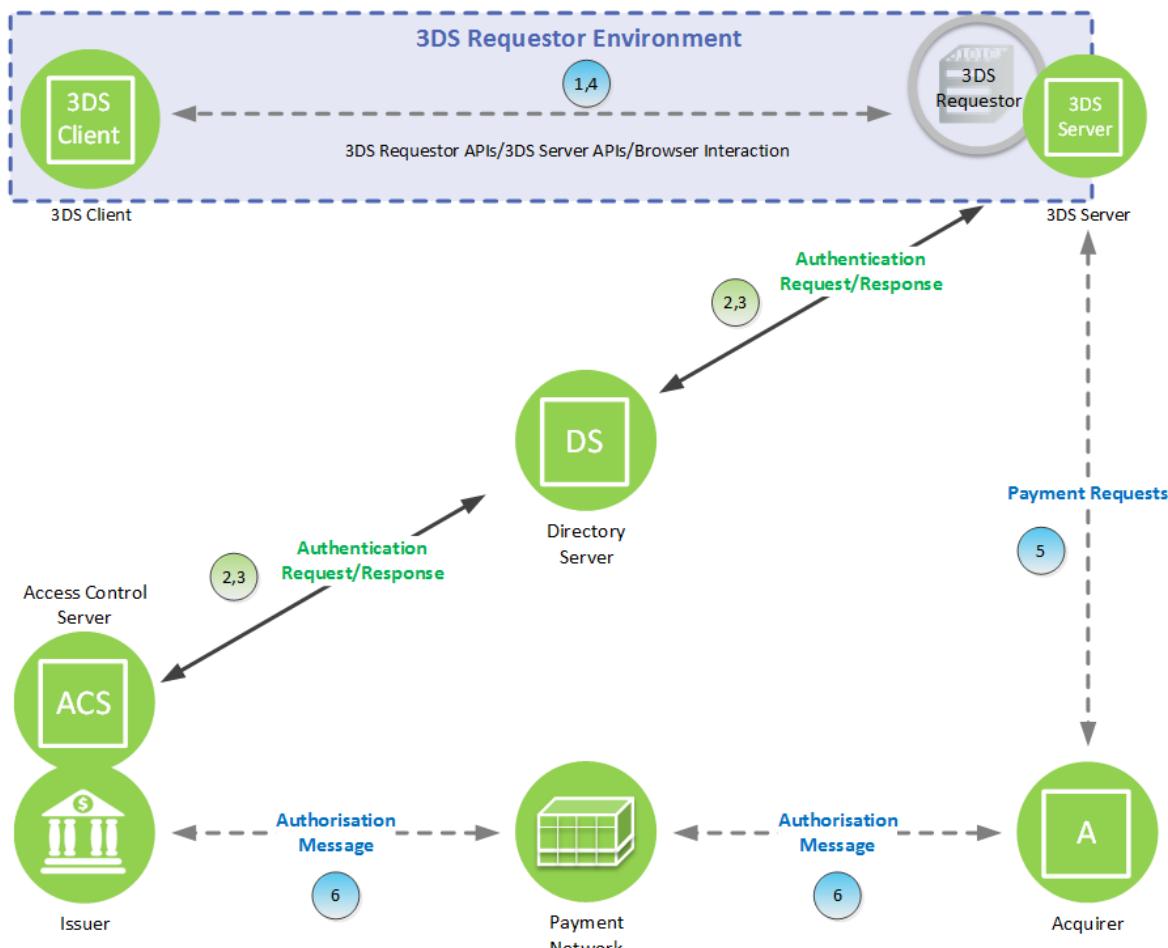
EMV Payment Token handling during transaction flow:

During the Authentication Request (AReq) message flow, it might be necessary for the Payment Token to be detokenised and the actual PAN to be placed in the Cardholder Account Number for the remainder of the AReq flow. For instance, this could be required if the transaction initiated with an EMV Payment Token and the ACS was configured based on the PAN. In this case, the EMV Payment Token Indicator in the AReq message is set to True to indicate that the transaction initiated with an EMV Payment Token.

2.6 Frictionless Flow Outline

Figure 2.2 depicts the steps of the Frictionless Flow.

Figure 2.2: Frictionless Flow



Note: Dashed arrows and 3DS Requestor are not part of 3DS specification and are shown for clarity only

The Frictionless Flow comprises the following Steps:

Start: Cardholder—Cardholder initiates a transaction on a Consumer Device. The Cardholder provides the information necessary for the authentication (Cardholder entry or already on file with the Merchant).

- 1. 3DS Requestor Environment**—Within the 3DS Requestor Environment, the necessary 3-D Secure information is gathered and provided to the 3DS Server for inclusion in the AReq message.

How information is provided, and from which component, depends on the following:

- Device Channel—App-based (Section 2.6.1) or Browser-based (Section 2.6.2)
- Message Category—Payment or Non-Payment
- 3DS Requestor 3-D Secure implementation (Section 2.1.1)

2. **3DS Server through DS to ACS**—Using the information provided by the Cardholder and data gathered within the 3DS Requestor Environment, the 3DS Server creates and sends an AReq message to the DS, which then forwards the message to the appropriate ACS.
3. **ACS through DS to 3DS Server**—In response to the AReq message, the ACS returns an ARes message to the DS, which then forwards the message to the initiating 3DS Server.

Before returning the response, the ACS evaluates the data provided in the AReq message. In a Frictionless Flow, the ACS determines that further Cardholder interaction is not required to complete the authentication.

4. **3DS Requestor Environment**—The 3DS Server communicates the result of the ARes message to the 3DS Requestor Environment which then informs the Cardholder.

As defined in Section 2.1.1, the 3DS Requestor determines how the interaction between these components is implemented. Refer to Sections 2.6.1 and 2.6.2 for additional information.

Note: 3-D Secure processing ends here. For Payment Authorisation, the subsequent steps apply:

5. **Merchant and Acquirer**—The Merchant proceeds with authorisation exchange with its Acquirer. If appropriate, the Merchant, Acquirer, or Payment Processor can submit a standard authorisation request.
6. **Payment Authorisation**—The Acquirer can process an authorisation with the Issuer through the Payment System and return the authorisation results to the Merchant.

2.6.1 3DS Requestor Environment—App-based

In an App-based model, the communication flows between the 3DS SDK/3DS Requestor App and the 3DS Server/3DS Requestor using the APIs made available from the 3DS Server/3DS Requestor.

Figure 2.3 depicts the 3DS Requestor Environment in an App-based model.

Figure 2.3: 3DS Requestor Environment—Frictionless Flow—App-based



Functionality for Step 1 and Step 4 is defined in Section 2.6, with the following clarifications:

Start: Cardholder and 3DS Requestor App—The Cardholder initiates a transaction using a 3DS Requestor App on a Consumer Device.

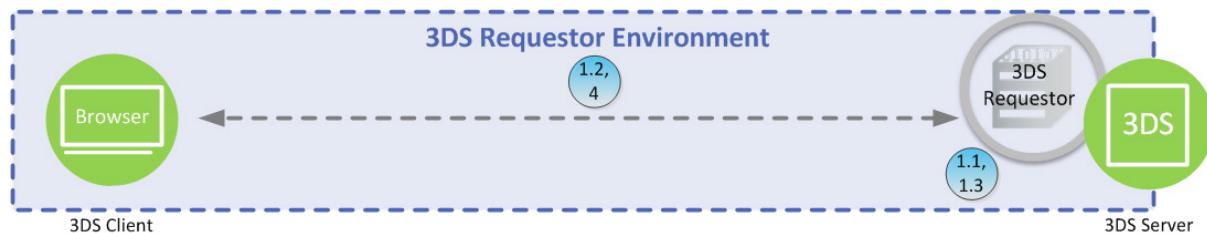
1. **3DS SDK and 3DS Server**—The 3DS SDK/3DS Requestor App communicates with the 3DS Server/3DS Requestor. Sensitive information from the Consumer Device is encrypted before being sent in the AReq message to the DS.

4. **3DS Server and 3DS SDK**—The 3DS Server/3DS Requestor communicates the result of the ARes message to the 3DS SDK/3DS Requestor App, which then informs the Cardholder.

2.6.2 3DS Requestor Environment—Browser-based

In a Browser-based model, the communication flows between the Consumer Device and the 3DS Server/3DS Requestor using a TLS Browser connection. Figure 2.4 depicts the 3DS Requestor Environment.

Figure 2.4: 3DS Requestor Environment—Browser-based



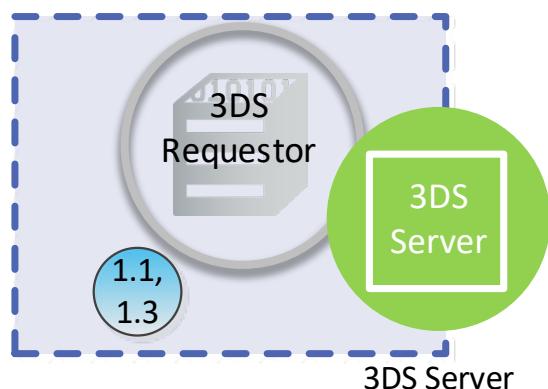
Functionality for Step 1 and Step 4 is defined in Section 2.6, with the following clarifications:
Start: Cardholder—The Cardholder initiates the transaction using a Browser on a Consumer Device using a website operated by the 3DS Requestor.

- 1.1 **3DS Requestor and 3DS Server**—The 3DS Requestor communicates with the 3DS Server. The 3DS Server determines the ACS and DS Protocol Version(s) and, if present obtains the 3DS Method URL for the requested card range and returns the information to the 3DS Requestor. The ACS and DS Protocol Version(s) and 3DS Method URL data were previously received by the 3DS Server via a PRes message.
- 1.2 **3DS Method on the 3DS Requestor checkout page**—The 3DS Requestor checkout page loads the 3DS Method URL, if present, which allows the ACS to obtain additional Browser information for risk-based decisioning.
- 1.3 **3DS Requestor and 3DS Server**—The 3DS Requestor provides the necessary 3-D Secure information for the transaction to the 3DS Server.
4. **3DS Server and 3DS Requestor**—The 3DS Server communicates the result of the ARes message to the 3DS Requestor and completes the transaction. The 3DS Integrator determines how the interaction between these components is implemented.

2.6.3 3DS Requestor Environment—3RI

For 3RI transactions, the 3DS Requestor Environment utilises only the 3DS Requestor and the 3DS Server as the transaction is initiated by the 3DS Requestor and not the cardholder. Figure 2.5 depicts the 3DS Requestor Environment for a 3RI transaction.

Figure 2.5: 3DS Requestor Environment—3RI



The communication flows between the 3DS Requestor and 3DS Server are defined by the specific 3DS Integrator implementation.

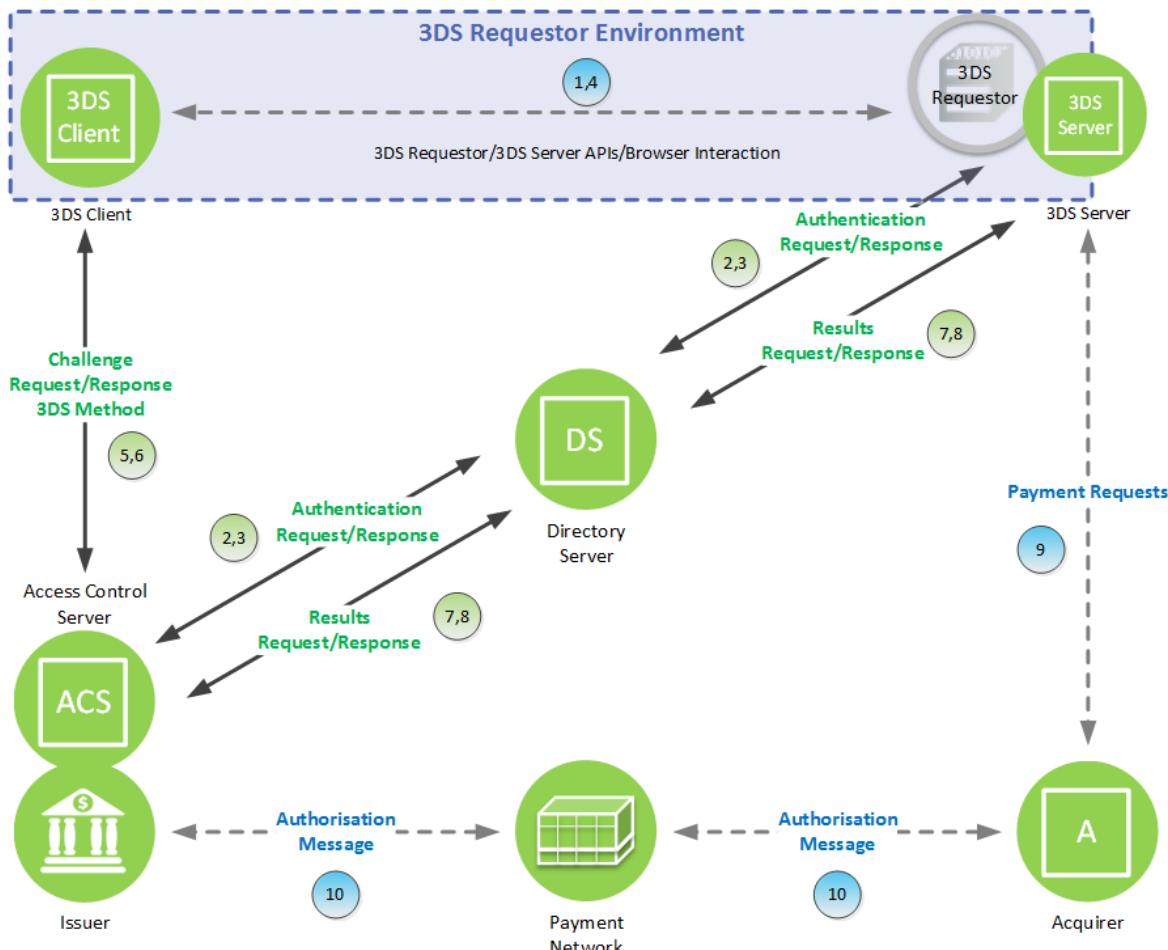
Start: 3DS Requestor—3DS Requestor needs to confirm an account.

- 1.1 **3DS Requestor and 3DS Server**—3DS Requestor initiates communications with the 3DS Server and provides the necessary 3-D Secure related information for the transaction to the 3DS Server.
- 1.3 **3DS Server and 3DS Requestor**—3DS Server communicates the result of the ARes message to the 3DS Requestor Server.

2.7 Challenge Flow Outline

Figure 2.6 depicts the steps of the Challenge Flow.

Figure 2.6: Challenge Flow



Note: Dashed arrows and 3DS Requestor are not part of 3DS specification and are shown for clarity only

The Challenge Flow comprises the following Steps:

Start: Cardholder—Same as the Frictionless Flow.

- 1. 3DS Requestor Environment**—Same as the Frictionless Flow.
- 2. 3DS Server through DS to ACS**—Same as the Frictionless Flow.
- 3. ACS through DS to 3DS Server**—Same as the Frictionless Flow except that the ARes message indicates that further Cardholder interaction is required to complete the authentication.
- 4. 3DS Server to 3DS Requestor Environment**—Same as the Frictionless Flow except that further Cardholder interaction is required to complete the authentication.

5. **3DS Client to ACS**—The 3DS Client initiates a CReq message based on information received in the ARes message. The manner in which this is done depends on the model:
 - **App-based**—A CReq message is formed by the 3DS SDK and is posted to the ACS URL received from the ARes message.
 - **Browser-based**—A CReq message is formed by the 3DS Server and is posted through the Cardholder Browser by the 3DS Requestor to the ACS URL received from the ARes message.
6. **ACS to 3DS Client**—The ACS receives the CReq message and interfaces with the 3DS Client to facilitate Cardholder interaction. The manner in which this is done depends on the model:
 - **App-based**—The ACS utilises pairs of CReq and CRes messages to perform the challenge. In response to the CReq message, the CRes message requesting the Cardholder to enter the authentication data is formed by the ACS and sent to the 3DS SDK.
 - **Browser-based**—The ACS sends the authentication user interface to the Cardholder Browser. The Cardholder enters the authentication data via the Browser to be checked by the ACS. In response to the CReq message, the CRes message is formed by the ACS and sent to the 3DS Server to indicate the result of the authentication.

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

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

Note: For Decoupled Authentication, instead of utilising the CReq and CRes messages, the ACS authenticates the Cardholder outside of the EMV 3-D Secure protocol.

7. **ACS through DS to 3DS Server**—The ACS sends an RReq message that can include the Authentication Value (AV) to the DS, which then routes the message to the appropriate 3DS Server using the 3DS Server URL received from the AReq message.
8. **3DS Server through DS to ACS**—The 3DS Server receives an RReq message and in response, returns an RRes message to the DS, which then routes the message to the ACS.

Note: 3-D Secure processing ends here. For Payment Authorisation, the subsequent steps apply.

9. **Merchant and Acquirer**—Same as the Frictionless Flow.
10. **Payment Authorisation**—Same as the Frictionless Flow.

3 EMV 3-D Secure Authentication Flow Requirements

This chapter provides the requirements for the EMV 3-D Secure processing flow. For clarity, the actions for all components involved in the 3-D Secure authentication process are described, but the four components covered by this *Core Specification* are:

- the 3DS Client
 - 3DS SDK
 - 3DS Method
- the 3DS Server
- the Directory Server (DS)
- the Access Control Server (ACS)

The components are required to follow the specifications as written.

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

As introduced in Chapter 2, a Challenge Flow is initially identical to a Frictionless Flow, thus the two flows in this section are described in one processing flow.

Note: The term “Validate” is used throughout this section and is a requirement for the 3-D Secure component to validate a received message. The details of the actual validation process are defined in a specific sub-section of Section 5.9. A validation process will include a check for the presence and format of each data element based on the data elements defined in Table A.1 and the functionality outlined in Section 5.1.6, and will also include the actual error handling, should a message be in error. In addition, the validation process may include cryptographic verification and decryption of the message.

3.1 App-based Requirements

The Steps in the authentication flow are outlined in Figure 3.1 with detailed requirements following the figure.

Figure 3.1: 3-D Secure Processing Flow Steps—App-based

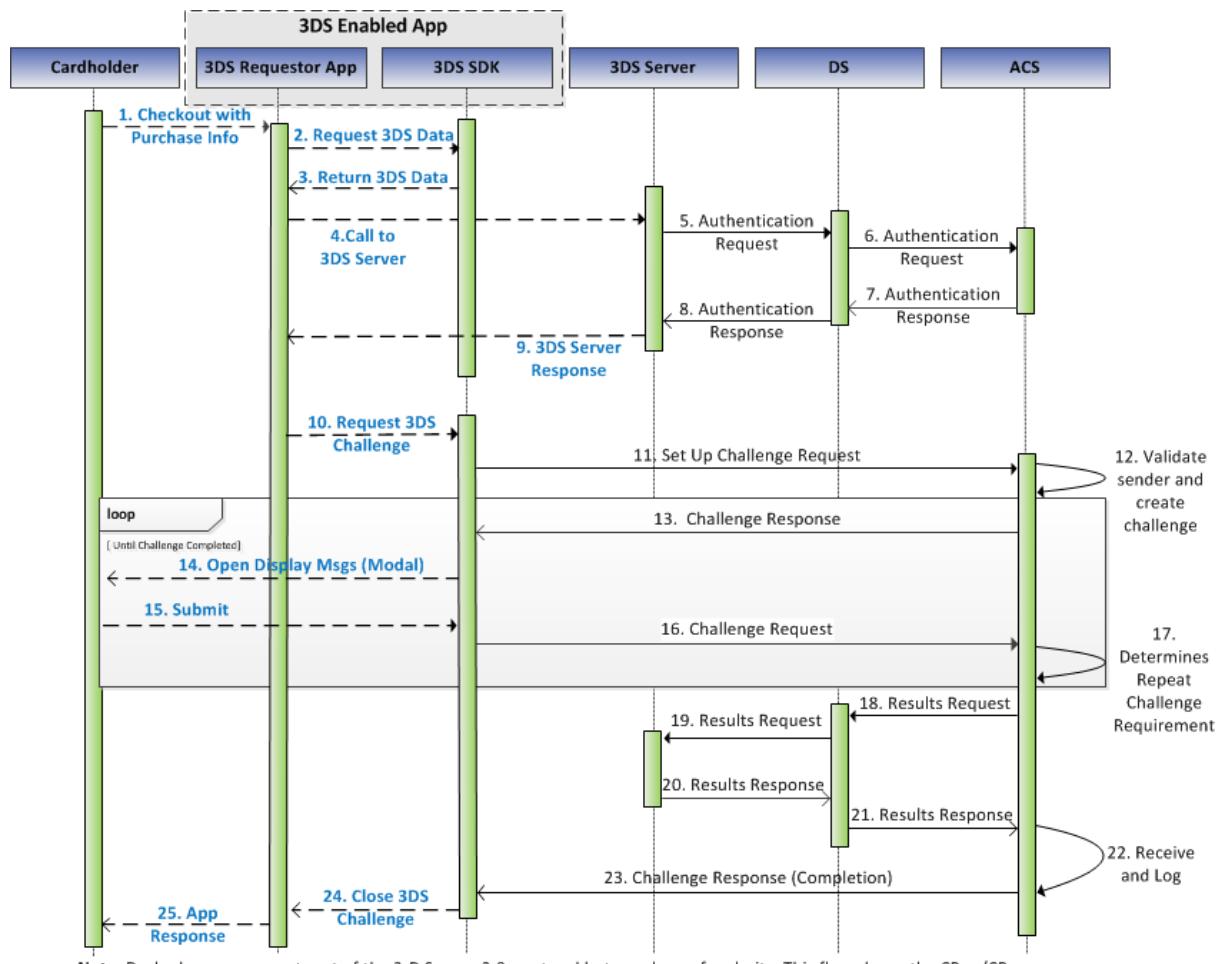


Figure 3.1 portrays a possible flow for components within the 3DS Requestor Environment and does not preclude a specific implementation. Refer to Section 2.1.1 for additional information about the 3DS Requestor Environment.

Note: Step 10 through Step 25 are applicable only for a Challenge Flow

Note: For a Decoupled Authentication challenge, instead of utilising the CReq and CRes messages (Steps 10 through 17 and Step 23), the ACS authenticates the Cardholder outside of the EMV 3-D Secure protocol. The Decoupled Authentication flow is portrayed in Figure 3.4 and outlined below.

The 3-D Secure processing flow for App-based implementations contains the following Steps:

Step 1: The Cardholder

The Cardholder interacts with the 3DS Requestor using the 3DS Requestor App and confirms the applicable business logic. For example, the Cardholder makes an e-commerce purchase using a 3DS Requestor App on a Consumer Device.

Step 2: The 3DS Requestor App

Depending on the 3DS Requestor Environment (as outlined in Section 2.1.1), additional information may be obtained. For example, payment and shopping cart information for Payment Authentication.

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

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

The 3DS Server shall:

- Seq 3.1 **[Req 11]** Determine which DS the authentication transaction needs to be sent based on the BIN (as defined in ISO 7812) and optionally other Cardholder account information.
- Seq 3.2 **[Req 419]** Use the protocol version lists from the ACS Protocol Versions and the DS Protocol Versions obtained from the PRes message and the protocol version supported by the 3DS SDK to set the highest common Message Version Number.

The 3DS Requestor App **activates the 3DS SDK** to initiate 3-D Secure Cardholder authentication.

The 3DS Requestor App provides to the 3DS SDK:

- the Directory Server ID value (which is the Payment System's RID), and
- the Message version (obtained from the 3DS Requestor Environment)

and obtains:

- the SDK Transaction ID
- the SDK App ID
- the Device Information, and
- the Protocol Version (Version used by the 3DS SDK for this transaction)

Note: If the message version is null, the 3DS SDK will utilise the highest version of the protocol that it supports. If the message version is present, the 3DS SDK will utilise that version of the protocol (assuming the 3DS SDK supports the version), otherwise the 3DS SDK will error.

Note: As described in the applicable 3DS SDK specification, the 3DS SDK encrypts the Device Information by using the DS public key. This key is identified based on the Directory Server ID that is passed when the 3DS SDK is activated.

Step 3: The 3DS SDK

The 3DS SDK returns data required for the AReq message.

Step 4: The 3DS Requestor Environment

The 3DS Requestor Environment is responsible for gathering the information for the AReq message assembled by the 3DS Server.

As introduced in Section 2.1.1, this specification does not require a specific component to gather the information only that the information is available for the 3DS Server when the AReq message is built. This information can include:

- Cardholder Account information
- Merchant Risk Indicator
- 3DS Requestor Authentication information
- Payment information (for Payment Authentication)
- Non-Payment information
- Cardholder information

This information together with the information gathered by the 3DS SDK (as outlined in the requirements within this step) is made available to the 3DS Server.

Note: UI requirements for this step are defined in Section 4.2.

Seq 3.3 **[Req 1]** The communication between the 3DS Requestor App and Server (3DS Requestor or 3DS Server) shall be established using a server authenticated TLS session as defined in section 6.1.1.

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

If the communication channel(s) within the 3DS Requestor Environment makes it impossible to have the 3DS Server receive the information within a reasonable amount of time, then this needs to be reported to the Cardholder via the 3DS Requestor App without further 3-D Secure processing.

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

Seq 3.4 **[Req 2]** Obtain the Device Information, SDK Reference Number, and SDK App ID. Refer to the applicable 3DS SDK specification and Annex A of this specification for additional detail.

Seq 3.5 **[Req 3]** Generate the SDK Transaction ID for the 3-D Secure authentication.

This ID will uniquely identify this transaction within all messages in the authentication process (AReq/ARes, CReq/CRes and RReq/RRes) for the 3DS SDK.

Seq 3.6 **[Req 4]** Set the Device Rendering Options Supported data element as defined in Table A.1.

Seq 3.7 **[Req 5]** Encrypt the Device Information (using the public key of the DS) as defined in Section 6.2.2.1.

Seq 3.8 **[Req 6]** Prepare the 3DS SDK to ACS secure channel as defined in Section 6.2.3.1.

Step 5: The 3DS Server

The 3DS Server shall:

Seq 3.9 **[Req 7]** Verify the authenticity of the 3DS SDK as defined in Section 6.2.1.

If the authenticity of the 3DS SDK cannot be verified by the 3DS Server, the 3DS Server **ends 3-D Secure processing**.

- Seq 3.10 **[Req 300]** If the 3DS Requestor and 3DS Server are separate components, ensure that data transferred between the components is protected at a level that satisfies Payment System security requirements with mutual authentication of both servers.
If the communication is not established as required, the 3DS Server **ends 3-D Secure processing**.
- Seq 3.11 **[Req 8]** Generate the 3DS Server Transaction ID.
- Seq 3.12 **[Req 9]** Obtain the 3DS Requestor ID, the 3DS Server Reference Number and conditionally, the Acquirer BIN.
- Seq 3.13 **[Req 10]** Ensure availability of the necessary information for the AReq message (as defined in Table B.1) gathered by components within the 3DS Requestor Environment.
If the necessary information is not available, the 3DS Server **ends 3-D Secure processing**.
- Seq 3.14 **[Req 12]** Establish a secure link with the DS as defined in Section 6.1.2.1.
If the connection cannot be established with the DS, the 3DS Server **ends 3-D Secure processing**.
If no PRes message information is available, then the 3DS Server may use a Message Version Number supported by the 3DS Server.
- Seq 3.15 **[Req 13]** Format the AReq message as defined in Table B.1.
- Seq 3.16 **[Req 14]** Send the AReq message to the DS using the secured link established in **[Req 12]**.
If the 3DS Server receives a failure when communicating with the DS or does not get a response (as defined in Section 5.5) the 3DS Server **ends 3-D Secure processing**.

Step 6: The DS

The DS shall:

- Seq 3.17 **[Req 15]** Receive the AReq message from the 3DS Server and Validate as defined in Section 5.9.1.
If the message is in error the DS **ends processing**.
- Seq 3.18 **[Req 390]** If SDK Type = 02, verify the signature in the SDK Server Signed Content, as defined in Section 6.2.2.4.
If the verification fails, the DS returns to the 3DS Server an Error Message (as defined in Section A.9) with Error Component = D and Error Code = 308 and **ends processing**.
- Seq 3.19 **[Req 394]** Determine if the SDK Type and the SDK Reference Number are valid for the transaction according to DS rules.
If not, the DS returns to the 3DS Server an Error Message (as defined in Section A.9) with Error Component = D and Error Code = 305 and **ends processing**.
- Seq 3.20 **[Req 16]** Generate the DS Transaction ID.
- Seq 3.21 **[Req 420]** Identify the DS public key used by the 3DS SDK to encrypt Device Information from the key identifier (kid) in the SDK Encrypted Data.

- If the DS detects an error with the key identifier, the DS returns to the 3DS Server an Error Message (as defined in Section A.9) with Error Component = D and Error Code = 311 and **ends processing**.
- Seq 3.22 **[Req 17]** Decrypt the SDK Encrypted Data data element of the AReq message as defined in Section 6.2.2.2 and Base64url encode resulting content and move the encoded content to the Device Information data element of the AReq message for the ACS.
- If decryption fails, the DS returns to the 3DS Server an Error Message (as defined in Section A.9) with Error Component = D and Error Code = 302 and **ends processing**.
- Seq 3.23 **[Req 18]** Check that the Message Version Number is supported by the DS and the ACS.
- If not, the DS returns to the 3DS Server EITHER:
- an ARes message (as defined in Table B.2) with Transaction Status set to the appropriate response as defined by the specific DS and **ends processing**, OR
 - an Error Message (as defined in Section A.9) with Error Component = D and Error Code = 102 and **ends processing**.
- Seq 3.24 **[Req 19]** Check the data elements in the AReq message as follows.
- If either:
 - the 3DS Server Reference Number does not represent a participating 3DS Server, OR
 - the SDK Reference Number does not represent a participating 3DS SDK
- then the DS returns to the 3DS Server an Error Message (as defined in Section A.9) with Error Component = D and Error Code = 303 and **ends processing**.
- If Merchant Category Code (MCC) is not valid for the specific DS, then the DS returns to the 3DS Server Error Message (as defined in Section A.9) with Error Component = D and Error Code = 306 and **ends processing**.
- Seq 3.25 **[Req 20]** Determine if the Cardholder Account Number received in the AReq message is in a participating account range.
- If not, the DS returns to the 3DS Server an ARes message (as defined in Table B.2 with the Transaction Status set to the appropriate value as defined by the specific DS and **ends processing**.
- Seq 3.26 **[Req 21]** Determine if the Cardholder Account Number is in an account range that has an ACS capable of processing 3-D Secure messages.
- If not, the DS returns to the 3DS Server an ARes message (as defined in Table B.2) with the Transaction Status set to the appropriate value as defined by the specific DS and **ends processing**.
- Seq 3.27 **[Req 22]** Store the 3DS Server URL with the DS Transaction ID (for possible RReq message processing).
- Seq 3.28 **[Req 23]** Establish a secure link with the ACS as defined in Section 6.1.3.1.
- If the connection cannot be established with the ACS then the DS proceeds as specified in **[Req 233]** and **ends processing**.

Note: The DS may maintain multiple ACS URLs. If the first URL attempted is not available, then the DS will attempt to connect to one of the alternate URLs.

Seq 3.29 **[Req 24]** Send the AReq message to the ACS using the secured link established in **[Req 23]**.

If the DS does not receive an ARes message from the ACS (as defined in Section 5.5), the DS returns to the 3DS Server a message as specified in **[Req 235]** and **ends processing**.

Step 7: The ACS

The ACS shall:

Seq 3.30 **[Req 25]** Receive the AReq message from the DS and Validate as defined in Section 5.9.2.

If the message is in error the ACS **ends processing**.

Seq 3.31 **[Req 26]** Check whether the Consumer Device is supported unless transaction will be processed as a Decoupled Authentication.

If device not supported, the ACS returns to the DS an ARes message with Transaction Status = U and Transaction Status Reason code = 03 and **ends processing**.

Seq 3.32 **[Req 386]** Check whether the Device Information data elements correspond to the Data Version Number.

If the Device Information data elements do not match the Data Version, the ACS returns an error message (Error Code = 203) or proceeds to process the transaction.

Note: The ACS uses the Device Information received in the AReq message to recognise the device, assess transaction risk, and determine if it can complete the authentication.

Note: When Decoupled Authentication is utilised, the Consumer Device that initiated the transaction does not need to be supported when the ACS has alternative approaches to authenticating the Cardholder.

Seq 3.33 **[Req 318]** The ACS shall not initiate an interaction with the Cardholder as part of a Frictionless transaction. Cardholder interaction shall be done as part of a Challenge Flow.

Seq 3.34 **[Req 27]** Generate the ACS Transaction ID.

Seq 3.35 **[Req 28]** Use the Cardholder Account Number from the AReq message to determine whether authentication is available or can be completed for the Cardholder.

If the authentication for the Cardholder Account Number is not available, then the ACS returns to the DS an ARes message (as defined in Table B.2) with Transaction Status and Transaction Status Reason code set to the appropriate response as defined by the specific DS and **ends processing**.

The ACS should:

Seq 3.36 **[Req 29]** Use the values of the 3DS Requestor Challenge Indicator, the 3DS Requestor Authentication Indicator and the 3DS Requestor Decoupled Request Indicator received in the AReq message when evaluating the transaction disposition as defined in **[Req 30]**.

The ACS shall:

- Seq 3.37 **[Req 30]** Evaluate the values received in the AReq message and determine whether the transaction¹ is:
- authenticated (Transaction Status = Y)
 - requiring a Cardholder challenge to complete authentication (Transaction Status = C)
 - requiring a Cardholder challenge using Decoupled Authentication (Transaction Status = D)
 - not authenticated (Transaction Status = N)
 - not authenticated, but a proof of authentication attempt (Authentication Value) was generated (Transaction Status = A)²
 - not authenticated, as authentication could not be performed due to technical or other issue (Transaction Status = U)
 - not authenticated because the Issuer is rejecting authentication and requesting that authorisation not be attempted (Transaction Status = R)
 - authentication not requested by the 3DS Server for data sent for informational purposes only (Transaction Status = I)

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

- Seq 3.38 **[Req 31]** If a transaction is deemed authenticated (Transaction Status = Y or A) the ACS performs the following:
- For a Payment Authentication (Message Category = 01-PA), the ECI value and Authentication Value shall be generated and included in the ARes message as defined by the DS.
 - For a Non-Payment Authentication (Message Category = 02-NPA), the ACS *may*:
 - Generate the ECI value and Authentication Value and include in the ARes message as defined by the DS.
 - Assign an appropriate Transaction Status Reason code value as defined by the specific DS and include in the ARes message.

Note: Whether the ECI Indicator/Authentication Value and/or the Transaction Status Reason value is included in the ARes message is defined by the specific DS.

- Seq 3.39 **[Req 32]** If a challenge is deemed necessary (Transaction Status = C), the ACS determines whether an acceptable challenge method is supported by the 3DS SDK based in part on the following data elements received in the AReq message: Device Channel, Device Rendering Options Supported, SDK Maximum Timeout and SDK Type. The ACS performs the following:
- a. Sets the Transaction Status = C for Challenge
 - b. Sets the ACS Rendering Type

¹ The decisioning process for this action is outside the scope of this specification.

² Support for Attempts is determined by each DS.

- c. Sets up the ACS to 3DS SDK secure channel (as defined in Section 6.2.3.2)
 - d. Stores the SDK Transaction ID (for subsequent CReq processing)
 - e. Stores the 3DS Server Transaction ID and DS Transaction ID (for subsequent RReq processing)
- Seq 3.40 **[Req 321]** If a Decoupled Authentication challenge is deemed necessary (Transaction Status = D) and 3DS Requestor Decoupled Request Indicator = Y or B, the ACS determines whether an acceptable challenge method is supported by the ACS based in part on the following data element received in the AReq message: 3DS Requestor Decoupled Max Time. The ACS performs the following:
- a. Sets Transaction Status = D for Decoupled Authentication.
 - b. Includes Decoupled (= 12) in the Authentication Method.
 - c. Sets ACS Decoupled Confirmation Indicator = Y.
 - d. Stores the 3DS Server Transaction ID and DS Transaction ID (for subsequent RReq processing).
- Seq 3.41 **[Req 33]** Complete formatting of the ARes message as defined in Table B.2.
- Seq 3.42 **[Req 34]** Send the ARes message to the DS using the secure link established in **[Req 23]**.
- Seq 3.43 **[Req 322]** For a Decoupled Authentication transaction (Transaction Status = D), do the following asynchronous process:
- a. Start timer against the 3DS Requestor Decoupled Max Time.
 - b. Authenticate the Cardholder. How an authentication decision is made is outside the scope of this specification. However, the ACS's objective is to complete the Cardholder authentication before the 3DS Requestor Decoupled Max Time expires.

Step 8: The DS

The DS shall:

- Seq 3.44 **[Req 305]** Check the data elements in the ARes message as follows:
- If the ACS Reference Number does not represent a participating ACS, the DS shall:
 - Return to the 3DS Server an Error Message (as defined in Section A.9) with Error Component = D and Error Code = 303 and **ends processing**, OR
 - Sends an ARes message (as defined in Table B.2) to the 3DS Server with Transaction Status set to the appropriate response as defined by the specific DS.
- Seq 3.45 **[Req 35]** Receive the ARes message or Error message from the ACS and Validate as defined in Section 5.9.3.
If the message is in error the DS **ends processing**.
- Seq 3.46 **[Req 36]** Log transaction information as required by the DS rules.

Seq 3.47 **[Req 421]** If the DS creates the ARes message on the ACS's behalf (for example, the DS returns a Transaction Status = A), then the DS sets the ACS Reference Number equal to the DS Reference Number and the ACS Transaction ID equal to the DS Transaction ID.

Seq 3.48 **[Req 37]** Send the ARes message to the 3DS Server received from the ACS using the secure link established in **[Req 12]**.

Step 9: The 3DS Server

The 3DS Server shall:

Seq 3.49 **[Req 38]** Receive the ARes message or Error Message from the DS and Validate as defined in Section 5.9.4.

If the message is in error the 3DS Server **ends processing**.

Seq 3.50 **[Req 39]** For an authenticated transaction (Transaction Status = Y or A):

- a. For Payment Authentication (Message Category = 01-PA), ensure that the Transaction Status, ECI value, and Authentication Value as generated by the ACS are provided for the authorisation process.
- b. Send necessary information from the ARes message (as defined in Table B.2) to the 3DS Requestor Environment.
- c. Continue with **[Req 79]**.

Seq 3.51 **[Req 40]** For a transaction with a challenge (Transaction Status = C):

Evaluate based in part on the 3DS Requestor Challenge Indicator, the ACS Challenge Mandated Indicator and the ACS Rendering Type whether to perform the requested challenge.

- If the 3DS Requestor accepts the challenge:
 - Send necessary information (as defined in Table B.2) from the ARes message to the 3DS Requestor Environment
 - Continue with Step 10
- If the 3DS Requestor continues without performing the requested challenge, receive the RReq message from the DS and Validate as defined in Section 5.9.9. If the message is in error, the 3DS Server **ends processing**. Format the RRes message as defined in Table B.9 and send to the DS. Further processing is outside the scope of 3-D Secure processing. The 3DS Server may continue with **[Req 79]** and **ends 3-D Secure processing**.

Seq 3.52 **[Req 41]** For a transaction not authenticated (Transaction Status = N, U, or R):

- a. Send necessary information (as defined in Table B.2) from the ARes message to the 3DS Requestor Environment.
- b. Continue with **[Req 79]**.

Seq 3.53 **[Req 323]** For a Decoupled Authentication transaction (Transaction Status = D), send necessary information (as defined in Table B.2) from the ARes message to the 3DS Requestor Environment.

Seq 3.54 **[Req 355]** Convey the Cardholder Information Text to the 3DS Requestor environment. The 3DS Requestor displays the Cardholder Information Text received to the Cardholder as depicted in Section A.20.

Note: The next step for a:

- **Frictionless Flow is [Req 79]**
- **Decoupled Authentication transaction is Step 18**
- **Challenge Flow is Step 10 (Step 10 through Step 23 and the first requirement of Step 24 are applicable only for a Challenge Flow [Transaction Status = C]).**

Note: For a Decoupled Authentication transaction, as defined in **[Req 345]**, the 3DS Server is now expected to wait for the ACS to authenticate the Cardholder at which time the ACS will send an RReq message.

Step 10 The 3DS Requestor App

The 3DS Requestor Environment receives the necessary ARes data elements from the 3DS Server and makes the data elements available to the 3DS SDK for execution of a Challenge Flow.

The 3DS Requestor App **performs the challenge** by making a call to the 3DS SDK.

Step 11: The 3DS Requestor Environment

The 3DS SDK shall:

- Seq 3.55 **[Req 42]** Check the received 3-D Secure data elements as defined in Table A.1.
- Seq 3.56 **[Req 43]** Complete the ACS to 3DS SDK secure channel as defined in Section 6.2.3.3.
If the secure channel cannot be completed, the 3DS SDK reports the error to the 3DS Requestor App and **ends 3-D Secure processing**.
- Seq 3.57 **[Req 44]** Establish a secure link to the ACS as defined in Section 6.1.4.1.
The link is established using the ACS URL received from the 3DS Server and verified as part of **[Req 43]**.
If the secure channel cannot be completed, the 3DS SDK reports the error to the 3DS Requestor App and **ends 3-D Secure processing**.
- Seq 3.58 **[Req 45]** Format the CReq message as defined in Table B.3 and protect the content as defined in Section 6.2.4.1.
If the content protection fails, the 3DS SDK reports the error to the 3DS Requestor App and **ends 3-D Secure processing**.
- Seq 3.59 **[Req 46]** Send the CReq message to the ACS using the secure link established in **[Req 44]**.

Step 12: The ACS

The ACS shall:

- Seq 3.60 **[Req 47]** Receive the CReq message or Error Message from the 3DS SDK and Validate as defined in Section 5.9.5.
If the message is in error the ACS **ends processing**.
- Seq 3.61 **[Req 48]** Set the Interaction Counter to zero.
- Seq 3.62 **[Req 49]** Set Challenge Completion Indicator = N.
- Seq 3.63 **[Req 50]** Obtain the information needed to display a Challenge on the Consumer Device per the selected challenge method and ACS UI Type. Refer to Section 4.2 for information about UI data elements.

Step 13: The ACS

The ACS shall:

- Seq 3.64 **[Req 51]** Format the CRes message as defined in Table B.4.
- Seq 3.65 **[Req 52]** Protect the content in the CRes message as defined in Section 6.2.4.4.
If the content protection fails, the ACS **ends 3-D Secure processing**.
- Seq 3.66 **[Req 53]** Send the CRes message to the 3DS SDK through the secure link established in **[Req 44]** for an initial interaction with the 3DS SDK, or **[Req 56]** for a continued interaction with the 3DS SDK.

Step 14: The 3DS SDK

The 3DS SDK shall:

- Seq 3.67 **[Req 54]** Receive the CRes message or Error Message from the ACS and Validate as defined in Section 5.9.7.
If the message is in error, the 3DS SDK **ends processing**.
- Seq 3.68 **[Req 55]** Display the UI based upon the ACS UI Type selected and the data elements populated. Refer to section 4.2 of the applicable 3DS SDK specification or for an OOB authentication (ACS UI Type = 04 or 06), refer to Section 3.2 for UI details.
If the CRes message for a Native UI contains a URL(s) directing the 3DS SDK to fetch data from an external server (i.e., an Issuer Image or Payment System Image for use with a Native UI), establish an additional secure link to the external server as defined in Section 6.1.4.1 and fetch and display the received data within the UI.
If a secure link cannot be established, the 3DS SDK proceeds with the challenge displays all other provided mandatory and optional UI data elements and does not send an error message to the ACS.

Step 15: The Cardholder Interaction with the 3DS SDK

The Cardholder interacts with the UI (for example, enters the data and selects Submit).

- Seq 3.69 **[Req 59]** If the Cardholder abandons the challenge during the processing of Step 12 through Step 15 the 3DS SDK sets the Challenge Cancelation Indicator to the appropriate value in the CReq message.

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

Step 16: The 3DS SDK

The 3DS SDK shall:

- Seq 3.70 **[Req 56]** Establish a secure link to the ACS as defined in Section 6.1.4.1.
- Seq 3.71 **[Req 57]** Format the CReq message as defined in Table B.3 and protect the contents as defined in Section 6.2.4.1.
If the content protection fails, the 3DS SDK reports the error to the 3DS Requestor App and **ends 3-D Secure processing**.
- Seq 3.72 **[Req 58]** Send one CReq message to the ACS using the secure link established in **[Req 56]** and wait for the ACS CRes message response before sending another CReq message.

Step 17: The ACS

The ACS shall:

- Seq 3.73 **[Req 60]** Receive the CReq message or Error Message from the 3DS SDK and Validate as defined in Section 5.9.5.
If the message is in error the ACS **ends processing**.
- Seq 3.74 **[Req 310]** If Challenge Cancelation Indicator has a value, then continue with Step 18.
- Seq 3.75 **[Req 461]** If the ACS determines that Decoupled Authentication Fallback is necessary and 3DS Requestor Decoupled Request Indicator = F or B, inform the Cardholder of Decoupled Authentication using the Information UI template as defined in Chapter 4 with additional CRes messages, then continue to **[Req 61]** to prepare the final CRes message.
- Seq 3.76 **[Req 61]** Check the data received in the CReq message and assess the status of the authentication.
- If the authentication is successful, then the ACS:
 - Increments the Interaction Counter
 - Sets Transaction Status = Y
 - Sets the ECI value as defined by the specific DS
 - Generates the Authentication Value as defined by the DS
 - Sets Challenge Completion Indicator = Y
 - Continues with Step 18
 - If the authentication has failed or is not completed, then the ACS:
 - Increments the Interaction Counter and compares it to the ACS maximum challenges.
 - If the Interaction Counter \geq ACS maximum challenges or the authentication has failed, the ACS:
 - Sets Transaction Status = N
 - Sets Transaction Status Reason = 19
 - Sets the ECI value as defined by the specific DS
 - Sets Challenge Completion Indicator = Y
 - Continues with Step 18
 - Else if the Interaction Counter $<$ ACS maximum challenges and the authentication is not completed, the ACS:
 - Obtains the information needed to display a repeat Challenge on the Consumer Device per the selected challenge method and ACS UI Type.
 - Continues with Step 13

Step 18: The ACS

The ACS shall, for all Challenge Flow transactions (ARes Transaction Status = C) and for Decoupled Authentication transactions (ARes Transaction Status = D), once the authentication as defined in **[Req 322].b** has completed or the timer as defined in **[Req 322].a** has expired, do the following:

- Seq 3.77 **[Req 462]** For a Challenge Flow (ARes Transaction Status = C), if 3DS Requestor Decoupled Request Indicator = F or B, and if the ACS has determined that Decoupled Authentication Fallback is necessary,
- Set Transaction Status = D.
 - Set Transaction Status Reason = 29 or 30.
- Seq 3.78 **[Req 62]** Format the RReq message as defined in Table B.8.
- Seq 3.79 **[Req 63]** Establish a secure link with the DS as defined in Section 6.1.3.2.
- Seq 3.80 **[Req 64]** If the Cardholder abandons the challenge during the processing of Step 16 and Step 17, or if the ACS receives an abandonment CReq message from the 3DS SDK (as defined in **[Req 60]**), then the ACS sets the Challenge Completion Indicator = Y in the CRes message and sets the Challenge Cancelation Indicator to the appropriate value in the RReq message. Refer to Annex A for the specific values.
- Seq 3.81 **[Req 65]** Send the RReq message to the DS using the secure link established in **[Req 63]**.
- Seq 3.82 **[Req 66]** Ensure that one RReq message is sent to the DS for each ARes message with a Transaction Status = C or D.
- Seq 3.83 **[Req 345]** Ensure for a Decoupled Authentication transaction that:
- An RReq message is sent immediately upon obtaining an authentication result (whether successful or not).
 - An RReq message without an authentication result (Transaction Status = U) is sent when the 3DS Requestor Decoupled Max Time expires—with a grace period of 1 hour.
- Note: It is recommended that an RReq message with Transaction Status = U contains Transaction Status Reason = 24 or 26 and Challenge Cancelation Indicator = 03.**

Step 19: The DS

The DS shall:

- Seq 3.84 **[Req 67]** Receive the RReq message from the ACS and Validate as defined in Section 5.9.8.
If the message is in error the DS **ends processing**.
- Seq 3.85 **[Req 68]** Establish a secure link with the 3DS Server as defined in Section 6.1.2.2 using the 3DS Server URL extracted from the AReq message and stored in **[Req 22]**.
- Seq 3.86 **[Req 69]** Send the RReq message to the 3DS Server using the secure link established in **[Req 68]**.

Step 20 The 3DS Server

The 3DS Server shall:

- Seq 3.87 **[Req 70]** Receive the RReq message or Error Message from the DS and Validate as defined in Section 5.9.9.
If the message is in error, the 3DS Server **ends processing**.
- Seq 3.88 **[Req 463]** If Transaction Status = D and if 3DS Requestor Decoupled Request Indicator = F or B, set Results Message Status to 04.

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

- Seq 3.89 **[Req 71]** Format the RRes message as defined in Table B.9 and send to the DS using the secure link established in **[Req 68]**.
- Seq 3.90 **[Req 346]** For a Decoupled Authentication transaction, at a minimum wait the specified 3DS Requestor Decoupled Max Time plus 1 hour, 30 seconds for the RReq message. If an RReq message is never received, further processing is outside the scope of 3-D Secure processing.

Note: If the RReq message is not received, then the 3DS Server should assume that the Decoupled Authentication is not successful.

Note: For Payment Authentication, the Merchant can now proceed with Authorisation processing with its Acquirer. However, the Merchant may first want to receive confirmation that the Cardholder has not abandoned the transaction.

Step 21 The DS

The DS shall:

- Seq 3.91 **[Req 72]** Receive the RRes message or Error Message from the 3DS Server and Validate as defined in Section 5.9.10.
If the message is in error the DS **ends processing**.
- Seq 3.92 **[Req 73]** Log transaction information as required by the DS.
- Seq 3.93 **[Req 74]** Send the RRes message to the ACS as received from the 3DS Server using the secure link established in **[Req 63]**.

Step 22 The ACS

The ACS shall:

- Seq 3.94 **[Req 75]** Receive the RRes message or Error Message from the DS and Validate as defined in Section 5.9.11.
If the message is in error the ACS **ends processing**.

Note: 3-D Secure processing completes for Decoupled Authentication transactions.

Step 23 The ACS

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

- Seq 3.95 **[Req 76]** Format the final CRes message (as defined in Table B.5) and protect the contents as defined in Section 6.2.4.4.
If the content protection fails, the ACS **ends 3-D Secure processing**.
- Seq 3.96 **[Req 77]** Send the final CRes message to the 3DS SDK using the secure link established in **[Req 56]**.

Step 24 The 3DS Requestor Environment

The 3DS SDK shall for a Challenge Flow transaction (ARes Transaction Status = C):

Seq 3.97 **[Req 78]** Receive the final CRes message or Error Message from the ACS and Validate as defined in Section 5.9.7.

If the message is in error, the 3DS SDK **ends processing**.

Seq 3.98 **[Req 79]** Convey the appropriate response to the 3DS Requestor App.

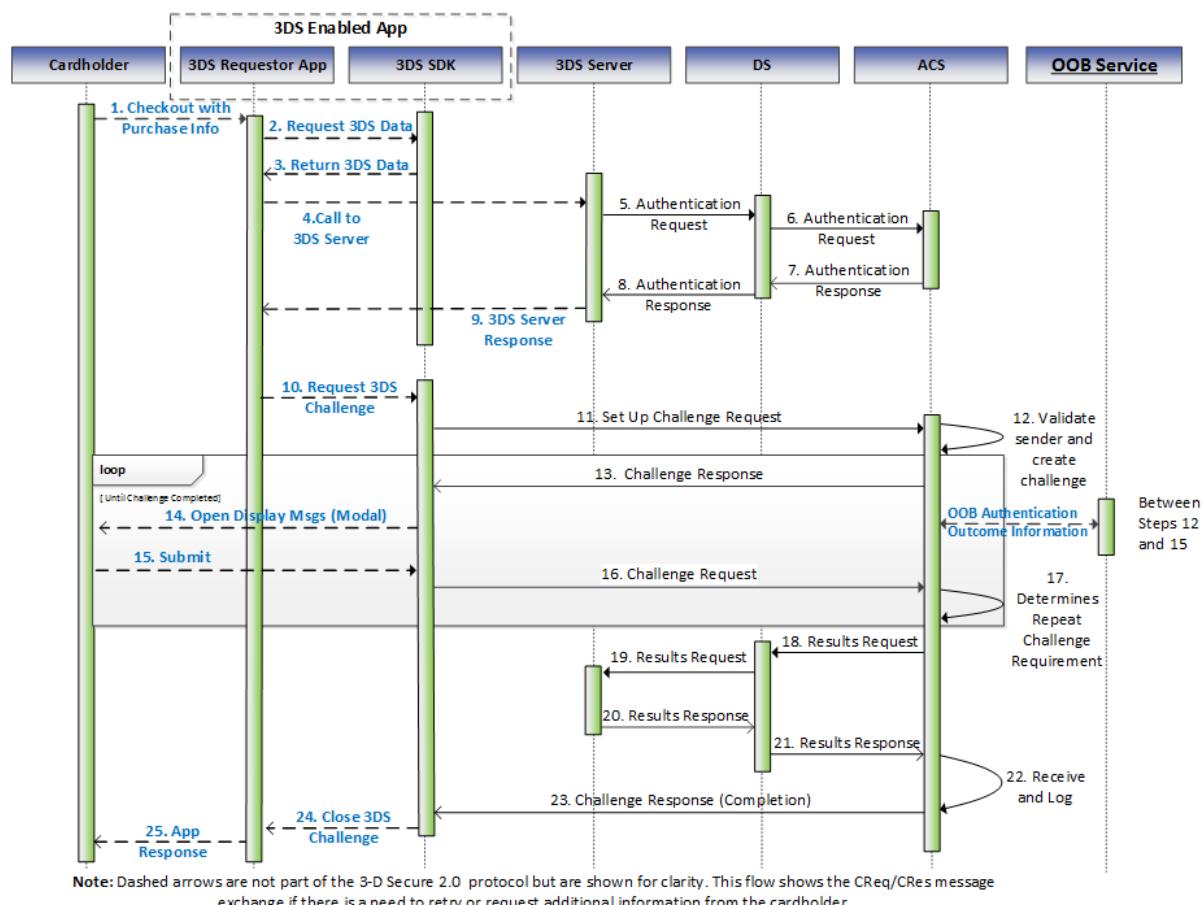
Note: 3-D Secure processing completes.

Step 25 The 3DS Requestor App

The 3DS Requestor App displays the appropriate result to the Cardholder.

3.2 Challenge Flow with OOB Authentication Requirements

Figure 3.2: Out-of-Band Processing Flow



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

- Step 7:** The ACS recognises that an OOB interaction with the Cardholder is required.
Step 13: The challenge information in the CRes message consists of Cardholder instructions on how to perform the OOB authentication.

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

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

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

Step 17: The ACS receives only an acknowledgement that the Cardholder may have performed the OOB authentication, thus in [Req 61] the ACS gathers the information on whether the authentication was successful from the OOB interaction with the Cardholder instead of the CReq message. If the ACS determines that the Cardholder did not

authenticate, then the ACS can update the Cardholder instructions through another CRes message.

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

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

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

3.2.1 OOB Requirements

This section defines additional requirements for an OOB flow.

Step 15 The Cardholder Interaction with the 3DS SDK

The 3DS SDK shall:

- Seq 3.99 **[Req 399]** For ACS UI Type = 04, set the OOB Continuation Indicator = 01 when the Cardholder selects the button with the OOB Continuation Label.
- Seq 3.100 **[Req 400]** For ACS UI Type = 04 or 06, if the 3DS Requestor App comes to the foreground, set the value of the OOB Continuation Indicator = 02, and continue automatically (without UI interaction by the Cardholder) with Step 16 in the App-based flow (send a CReq message to the ACS).

3.2.2 OOB Automatic Switching Features

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

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

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

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

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

Step 13: The ACS

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

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

- Seq 3.101 **[Req 401]** For ACS UI Type = 04 or 06, set the OOB App URL to the URL value used during installation of the OOB Authentication App.

Seq 3.102 **[Req 402]** For ACS UI Type = 06, include in the OOB challenge HTML code an action that triggers a location change to the `HTTPS://EMV3DS/openoobApp` URL when the Cardholder selects the button.

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

Step 14: The 3DS SDK

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

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

Seq 3.103 **[Req 403]** For ACS UI Type = 04, display a button with the OOB App Label used for the switch to the OOB Authentication App.

Step 15: The Cardholder Interaction with the 3DS SDK

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

3.2.2.1 OOB App URL Requirements

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

Seq 3.104 **[Req 404]** For ACS UI Type = 04, attempt to open the OOB Authentication App by using the OOB App URL when the Cardholder selects the button with the OOB App Label.

Seq 3.105 **[Req 405]** For ACS UI Type = 06:

- a. Intercept a location change event that is sent to the specific `HTTPS://EMV3DS/openoobApp` URL.
- b. Attempt to open the OOB Authentication App by using the OOB App URL.

Seq 3.106 **[Req 406]** If the attempt to open the OOB Authentication App is successful, then continue with Section 3.2.2.2.

Seq 3.107 **[Req 407]** If the attempt to open the OOB Authentication App fails (for example, the platform method to open the OOB App URL returns an error), then:

- a. Set the OOB App Status to the appropriate value as defined in Table A.1.
- b. Set the OOB Continuation Indicator = 02.
- c. Continue automatically (without UI interaction by the Cardholder) with Step 16 in the App flow.

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

3.2.2.2 3DS Requestor App URL

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

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

The 3DS SDK shall:

- Seq 3.108 **[Req 408]** Display the UI template and data elements received in the last CRes message.
- Seq 3.109 **[Req 409]** For ACS UI Type = 04 or 06, set the OOB Continuation Indicator = 02 and continue with Step 16 in the App-based flow.

3.3 Browser-based Requirements

The Steps in the authentication flow are outlined in Figure 3.3 with detailed requirements following the figure.

Figure 3.3: 3-D Secure Processing Flow Steps—Browser-based

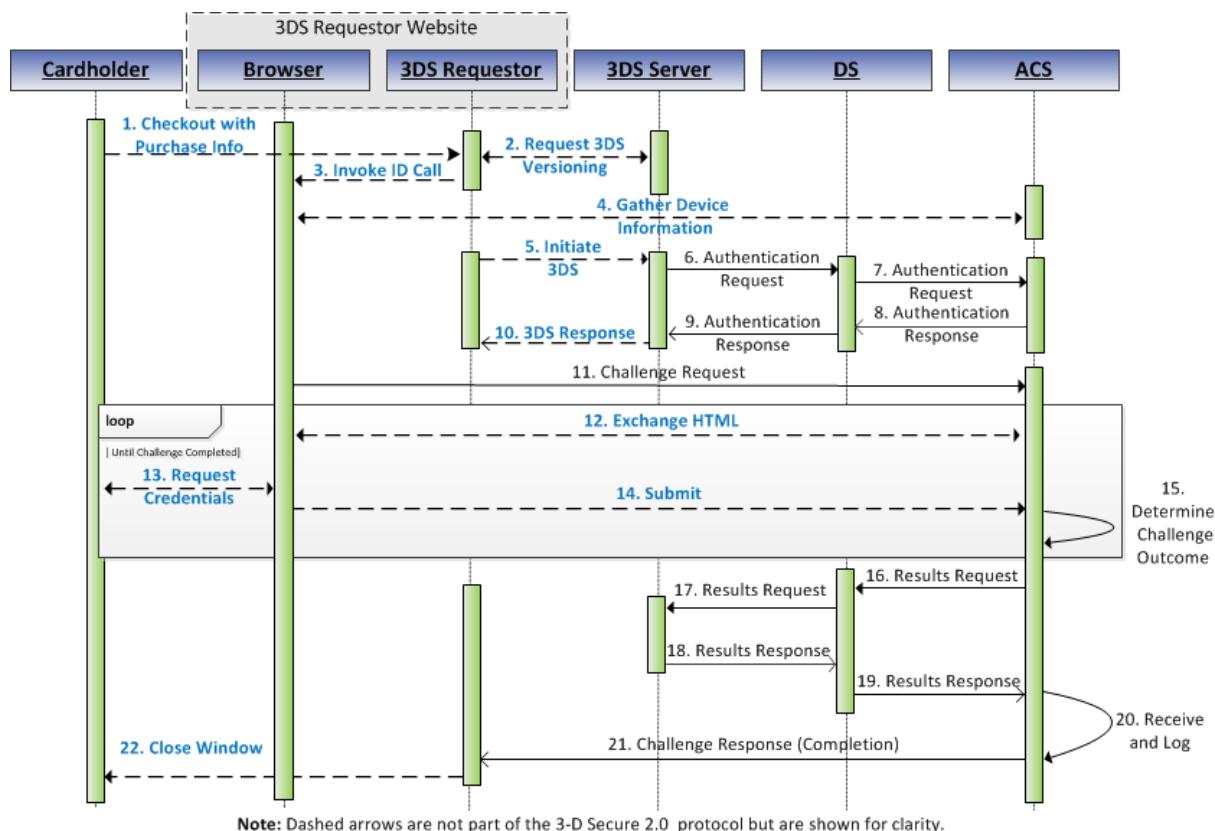


Figure 3.3 portrays a possible flow for components within the 3DS Requestor Environment and does not preclude a specific implementation. Refer to Section 2.1.1 for additional information about the 3DS Requestor Environment.

Note: Step 10 through Step 21 are applicable only for the Challenge Flow.

Note: For a Decoupled Authentication challenge, instead of utilising the CReq and CRes messages (Steps 11 through 15 and Step 21), the ACS authenticates the Cardholder outside of the EMV 3-D Secure protocol. The Decoupled Authentication flow is portrayed in Figure 3.4 and outlined below.

The 3-D Secure processing flow for Browser-based implementations contains the following Steps:

Step 1: The Cardholder

The Cardholder interacts with the 3DS Requestor using a Browser on a Consumer Device and confirms the applicable business logic. For example, the Cardholder makes an e-commerce purchase on a Merchant website using a Consumer Device.

Step 2: The 3DS Server/3DS Requestor

The 3DS Requestor initiates communications with the 3DS Server and provides the necessary 3-D Secure information to the 3DS Server to initiate Cardholder authentication.

Depending on the 3DS Requestor Environment (as outlined in Section 2.6.2) additional information may be obtained.

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

The 3DS Server shall:

Seq 3.110 **[Req 80]** Retrieve the ACS Protocol Version and DS Protocol Version lists and if present, the 3DS Method URL (stored from a previously received PRes message) for the card range.

Seq 3.111 **[Req 81]** Generate the 3DS Server Transaction ID.

Seq 3.112 **[Req 82]** Pass the 3DS Server Transaction ID, ACS Protocol Versions, DS Protocol Versions and if present, the 3DS Method URL back through the 3DS Requestor Environment to the 3DS Requestor.

If the DS Protocol Versions is not present for the card range, then the default values for the DS Protocol Versions located in the PRes message shall be utilised.

Step 3 The 3DS Requestor Environment

The 3DS Server shall:

Seq 3.113 **[Req 83]** For each transaction ensure that the 3DS Server Transaction ID used in the 3DS Method on the 3DS Requestor website is the same 3DS Server Transaction ID used in the AReq message.

Seq 3.114 **[Req 84]** Ensure that the 3DS Method is executed on the 3DS Requestor website if a 3DS Method URL exists as defined in Section 5.8.1.

Step 4 Browser and the ACS

If a 3DS Method URL was present in the response from the 3DS Server in Step 2, the Browser will connect via the 3DS Method to the ACS or an entity designated by the ACS to gather Browser and Device Information.

The manner in which the 3DS Method obtains Device Information and which information is gathered is outside the scope of this specification. However, it is necessary to use the 3DS Server Transaction ID to identify the Browser/Device Information for a later match at the ACS. Refer to Chapter 5 for additional information about message handling.

The manner in which the Device Information is retrieved or used by the ACS or an entity designated by the ACS is outside the scope of this specification. However, it is a requirement that the Browser connects to the ACS using a secure link (as defined in Section 6.1.8).

The ACS shall:

Seq 3.115 **[Req 85]** Ensure that the communication between the Browser and the ACS is established using a server authenticated TLS session (as defined in Section 6.1.8).

If the communication is not established as required, the ACS **ends processing**.

Note: The 3DS Method Requirements are defined in Section 5.8.1.

Step 5: The 3DS Requestor Environment

The 3DS Requestor Environment is responsible for gathering the information for the AReq message assembled by the 3DS Server.

As introduced in Section 2.1.1, this specification does not require a specific component to gather the information, only that the information is available for the 3DS Server when the AReq message is built. This information can include:

- Cardholder Account Information
- Merchant Risk Indicator
- 3DS Requestor Authentication Information
- 3DS Requestor Prior Transaction Authentication Information
- Payment Information
- Non-Payment Information
- Cardholder information

This information is made available to the 3DS Server.

The 3DS Server shall:

- Seq 3.116 **[Req 86]** Ensure that the communication between client (Browser) and server (3DS Requestor) has been established using a server authenticated TLS session as defined in Section 6.1.1.
- If the communication is not established as required, the 3DS Server **ends 3-D Secure processing**.
 - If the communication channel(s) within the 3DS Requestor Environment makes it impossible to have the 3DS Server receive the information within a reasonable amount of time, then this needs to be reported to the Cardholder via the Browser and **ends 3-D Secure processing**.

- Seq 3.117 **[Req 301]** Ensure, if the 3DS Requestor and 3DS Server are separate components, that data transferred between the components is protected at a level that satisfies Payment System security requirements with mutual authentication of both servers.

If the communication is not established as required, the 3DS Server **ends 3-D Secure processing**.

Step 6 The 3DS Server

The 3DS Server shall:

- Seq 3.118 **[Req 87]** Obtain the 3DS Requestor ID, the 3DS Server Reference Number, and conditionally the Acquirer BIN.

- Seq 3.119 **[Req 88]** Ensure availability of the necessary information for the AReq message (as defined in Table B.1) gathered by components within the 3DS Requestor Environment.

If the necessary information is not available, the 3DS Server **ends 3-D Secure processing**.

- Seq 3.120 **[Req 441]** Ensure that the 3DS Requestor executed the 3DS Method within the previous 10 minutes. Otherwise, the 3DS Requestor re-executes the 3DS Method as defined in Section 5.8.1.

- Seq 3.121 **[Req 89]** Determine which DS the authentication transaction needs to be sent based on the BIN (as defined in ISO 7812) and optionally other Cardholder account information.

- Seq 3.122 **[Req 90]** Establish a secure link with the DS as defined in Section 6.1.2.1.

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

- Seq 3.123 **[Req 422]** Use the protocol version lists from the ACS Protocol Versions and DS Protocol Versions obtained from the PRes message to set the highest common Message Version Number.
- If no PRes message information is available, then the 3DS Server may use a Message Version Number supported by the 3DS Server.
- Seq 3.124 **[Req 91]** Format the AReq message as defined in Table B.1.
- Seq 3.125 **[Req 92]** Send the AReq message to the DS using the secured link established in **[Req 90]**.
- If the 3DS Server receives a failure when communicating with the DS or does not get a response (as defined in Section 5.5) then the 3DS Server **ends 3-D Secure processing**.

Step 7 The DS

The DS shall:

- Seq 3.126 **[Req 93]** Receive the AReq message from the 3DS Server and Validate as defined in Section 5.9.1.
- If the message is in error the DS **ends processing**.
- Seq 3.127 **[Req 94]** Generate the DS Transaction ID.
- Seq 3.128 **[Req 95]** Check that the Message Version Number is supported by the DS and the ACS.
- If not, the DS returns to the 3DS Server EITHER:
- an ARes message (as defined in Table B.2) with Transaction Status set to the appropriate response as defined by the specific DS and **ends processing**, OR
 - an Error Message (as defined in Section A.9) with Error Component = D and Error Code = 102 and **ends processing**.
- Seq 3.129 **[Req 96]** Further check the data elements in the AReq message as follows:
- If the 3DS Server Reference Number does not represent a participating 3DS Server, then the DS returns to the 3DS Server an Error Message (as defined in Section A.9) with Error Component = D and Error Code = 303 then **ends processing**.
 - If Merchant Category Code (MCC) is not valid for the specific DS, then the DS returns to the 3DS Server an Error Message (as defined in Section A.9) with Error Component = D and Error Code = 306 and **ends processing**.
- Seq 3.130 **[Req 97]** Determine if the Cardholder Account Number received in the AReq message is in a participating account range.
- If not, the DS returns to the 3DS Server an ARes message (as described in Table B.2) with the Transaction Status set to the appropriate value as defined by the specific DS and **ends processing**.

Seq 3.131 **[Req 98]** Determine if the Cardholder Account Number is in an account range that has an ACS capable of processing 3-D Secure messages.

If not, the DS returns to the 3DS Server an ARes message (as described in Table B.2) with the Transaction Status set to the appropriate value as defined by the specific DS and **ends processing**.

Seq 3.132 **[Req 99]** Store the 3DS Server URL with the DS Transaction ID (for possible RReq processing).

Seq 3.133 **[Req 100]** Establish a secure link with the ACS as defined in Section 6.1.3.1.

If the connection cannot be established with the ACS then the DS proceeds as specified in **[Req 233]** and **ends processing**.

Note: The DS may maintain multiple ACS URLs. If the first URL attempted is not available, then the DS will attempt to connect to one of the alternate URLs.

The DS shall:

Seq 3.134 **[Req 101]** Send the AReq to the ACS using the secured link established in **[Req 100]**.

If the DS does not receive an ARes message from the ACS (as defined in Section 5.5), the DS returns to the 3DS Server a message as specified in **[Req 235]** and **ends processing**.

Step 8 The ACS

The ACS shall:

Seq 3.135 **[Req 102]** Receive the AReq message from the DS and Validate as defined in Section 5.9.2.

If the message is in error the ACS **ends processing**.

Seq 3.136 **[Req 103]** Check whether the Consumer Device on which the authentication is being requested is supported.

If not, the ACS returns to the DS an ARes message with a Transaction Status = U and Transaction Status Reason code = 03 and **ends processing**.

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

Seq 3.137 **[Req 410]** Retrieve the data from a previous 3DS Method execution if the 3DS Method ID is present.

Seq 3.138 **[Req 319]** The ACS shall not initiate an interaction with the Cardholder as part of a Frictionless transaction. Cardholder interaction shall be done as part of a Challenge Flow.

Seq 3.139 **[Req 104]** Generate the ACS Transaction ID.

Seq 3.140 **[Req 105]** Use the Cardholder Account Number from the AReq message to determine whether authentication is available or can be completed for the Cardholder.

If the authentication for the Cardholder Account Number is not available, the ACS returns to the 3DS Server an ARes message (as defined in Table B.2) with the Transaction Status, and the Transaction Status Reason code set to the appropriate response as defined by the specific DS and **ends processing**.

The ACS should:

Seq 3.141 **[Req 106]** Use the values of the 3DS Requestor Challenge Indicator, the 3DS Requestor Authentication Indicator and the 3DS Requestor Decoupled Request Indicator received in the AReq message when evaluating the transaction disposition as defined in **[Req 107]**.

The ACS shall:

Seq 3.142 **[Req 107]** Evaluate the values received in the AReq message and determine whether the transaction³ is:

- authenticated (Transaction Status = Y)
- requiring a Cardholder challenge to complete authentication (Transaction Status = C)
- requiring a Cardholder challenge using Decoupled Authentication (Transaction Status = D)
- not authenticated (Transaction Status = N)
- not authenticated, but a proof of authentication attempt (Authentication Value) was generated (Transaction Status = A)⁴
- not authenticated because authentication could not be performed due to a technical or other problem (Transaction Status = U)
- not authenticated because the Issuer is rejecting authentication and requesting that authorisation not be attempted (Transaction Status = R)
- authentication not requested by the 3DS Server for data sent for informational purposes only (Transaction Status = I)
- requiring an SPC authentication (Transaction Status = S). See Section 3.5.1 for details

Seq 3.143 **[Req 108]** If a transaction is deemed authenticated (Transaction Status is = Y or A), the ACS performs the following:

- For a Payment Authentication (Message Category = 01), the ECI value and Authentication Value shall be generated and included in the ARes message as defined by the DS.
- For a Non-Payment Authentication (Message Category = 02), the ACS may:
 - Generate the ECI value and Authentication Value and include in the ARes message as defined by the specific Payment System.

³ The decisioning process for this action is outside the scope of this specification.

⁴ Support for attempts is determined by each DS.

- Assign an appropriate Transaction Status Reason value as defined by the specific DS and include in the ARes message.

Note: Whether the ECI Indicator/Authentication Value and/or the Transaction Status Reason value is included in the ARes message is defined by the specific DS.

Seq 3.144 **[Req 109]** If a challenge is deemed necessary (Transaction Status = C), the ACS determines whether an acceptable challenge method is supported by the 3DS Server considering the 3DS Requestor Challenge Indicator data element received in the AReq message. The ACS performs the following:

- a. Sets the Transaction Status = C for Challenge.
- b. Sets the ACS URL field in the ARes message that will be utilised in the Browser to ACS link.
- c. Stores the 3DS Server Transaction ID, DS URL, and DS Transaction ID (for subsequent RReq processing).

Seq 3.145 **[Req 325]** If a Decoupled Authentication challenge is deemed necessary (Transaction Status = D) and 3DS Requestor Decoupled Request Indicator = Y or B, the ACS determines whether an acceptable challenge method is supported by the ACS based in part on the following data element received in the AReq message: 3DS Requestor Decoupled Max Time. The ACS performs the following:

- a. Sets the Transaction Status = D for Decoupled Authentication.
- b. Sets the ACS Decoupled Confirmation Indicator = Y.
- c. Stores the 3DS Server Transaction ID and DS Transaction ID (for subsequent RReq processing).

Seq 3.146 **[Req 110]** Complete formatting of the ARes message as defined in Table B.2.

Seq 3.147 **[Req 111]** Send the ARes message to the DS using the secure link established in **[Req 100]**.

Seq 3.148 **[Req 326]** For a Decoupled Authentication transaction, (Transaction Status = D), do the following asynchronous process:

- a. Start timer against the 3DS Requestor Decoupled Max Time.
- b. Authenticate the cardholder. How an authentication decision is made is outside the scope of this specification. However, the ACS's objective is to complete the Cardholder authentication before the 3DS Requestor Decoupled Max Time expires.

Step 9 The DS

The DS shall:

Seq 3.149 **[Req 306]** Check the data elements in the ARes message as follows.

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

- Seq 3.150 **[Req 112]** Receive the ARes message or Error message from the ACS and Validate as defined in Section 5.9.3.
If the message is in error the DS **ends processing**.
- Seq 3.151 **[Req 113]** Log transaction information as required by the DS rules.
- Seq 3.152 **[Req 411]** If the DS creates the ARes message on the ACS's behalf (for example, the DS returns a Transaction Status = A), then set the ACS Reference Number equal to the DS Reference Number and the ACS Transaction ID equal to the DS Transaction ID.
- Seq 3.153 **[Req 114]** Send the ARes message to the 3DS Server received from the ACS using the secure link established in **[Req 100]**.

Step 10 The 3DS Server

The 3DS Server shall:

- Seq 3.154 **[Req 115]** Receive the ARes message or Error Message from the DS and Validate as defined in Section 5.9.4.
If the message is in error the 3DS Server **ends processing**.
- Seq 3.155 **[Req 116]** For an authenticated transaction (Transaction Status = Y or A):
 - For a Payment Authentication (01-PA), ensure that the Transaction Status, ECI value, and Authentication Value as generated by the ACS are provided for the authorisation process.
 - Send necessary information (as defined in Table B.2) from the ARes message to the 3DS Requestor Environment.
 - Continue with Step 22.
- Seq 3.156 **[Req 117]** For a transaction with a challenge (Transaction Status = C):
 - Evaluate based in part on the 3DS Requestor Challenge Indicator, the ACS Challenge Mandated Indicator and the ACS Rendering Type whether to perform the requested challenge.
 - If the 3DS Requestor accepts the challenge:
 - Send necessary information (as defined in Table B.2) from the ARes message to the 3DS Requestor Environment.
 - Continue with step b through e of this requirement and then Step 11.
 - If the 3DS Requestor continues without performing the requested challenge, receive the RReq message from the DS and Validate as defined in Section 5.9.9. If the message is in error, the 3DS Server **ends processing**. Format the RRes message as defined in Table B.9 and send to the DS. Further processing is outside the scope of 3-D Secure processing. The 3DS Server may continue with Step 22.
 - Format the CReq message according to the format specified in Table B.3 for a Browser-based implementation.
 - Base64url encode the CReq message.
 - Construct a form containing the CReq message, and if provided by the 3DS Requestor, the 3DS Requestor Session Data (as defined in Table A.3).

- e. Pass the CReq message through the Cardholder Browser as defined in Section 5.8.2 to the ACS URL received in the ARes message, by causing the Cardholder Browser to POST the form to the ACS URL using a server authenticated TLS link as defined in Section 6.1.4.2.
- Seq 3.157 **[Req 118]** For a transaction not authenticated (Transaction Status = N, U, or R):
 - a. Send necessary information (as defined in Table B.2) from the ARes message to the 3DS Requestor Environment.
 - b. Continue with Step 22.
- Seq 3.158 **[Req 327]** For a Decoupled Authentication transaction (Transaction Status = D), send necessary information (as defined in Table B.2) from the ARes message to the 3DS Requestor Environment.

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

- Seq 3.159 **[Req 356]** Convey the Cardholder Information Text to the 3DS Requestor environment. The 3DS Requestor displays the Cardholder Information Text received to the Cardholder as depicted in Section A.20.

Note: *[Req 117].d specifies posting the CReq message from the 3DS Server through the Cardholder Browser to the ACS. This flow is only partially depicted in Step 10 of Figure 3.3.*

Note: ACS implementations that use JavaScript for redirection will also need to support a fallback for environments that do not support JavaScript as defined in *[Req 324]*.

Note: The next step for:

- Frictionless Flow is Step 22.
- Decoupled Authentication transaction is Step 16
- Challenge Flow is Step 11 (Step 11 through Step 21 are applicable only for a Challenge Flow [Transaction Status = C]).

Note: For a Decoupled Authentication transaction, as defined in *[Req 347]*, the 3DS Server is now expected to wait for the ACS to authenticate the Cardholder at which point in time the ACS will send an RReq message.

Step 11 The ACS

The ACS shall:

- Seq 3.160 **[Req 119]** Receive the CReq message from the Browser and Validate as defined in Section 5.9.6.
If the message is in error, the ACS **ends processing**.
- Seq 3.161 **[Req 120]** Prepare the authentication User Interface (ACS UI) to the Cardholder Browser.
- Seq 3.162 **[Req 121]** Set the Interaction Counter to zero.
- Seq 3.163 **[Req 442]** If the ACS receives more than one CReq message, the ACS either:
 - Restarts or continues the challenge with the Cardholder, OR
 - Returns an Error Message if it is not possible to continue or restart the authentication.

Step 12 The ACS and Browser

The ACS shall:

- Seq 3.164 **[Req 307]** The ACS shall not lead the Cardholder outside of the authentication flow by redirecting to any registration or marketing pages. Any redirection shall be used for authentication purposes only and within the iframe. The ACS shall only load external resources that are needed to improve the cardholder authentication experience and security (e.g., logos).
- Seq 3.165 **[Req 122]** Send the ACS UI to the Cardholder over the channel established by the HTTP POST in Step 10. The ACS shall allow the content of the UI to be framed. The Browser displays the ACS UI to the Cardholder.

Note: An Out-of-Band (OOB) Challenge Flow is identical to a standard 3-D Secure Processing Flow for a challenge for the Browser channel.

The ACS UI consists of Cardholder instructions on how to perform the OOB authentication.

The ACS initiates an OOB interaction with the Cardholder rather than interacting with the Cardholder via the Browser challenge iframe. During the OOB authentication the Cardholder authenticates to the ACS or a service provider/Issuer interacting with the ACS.

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

Step 13 The Cardholder

The Cardholder enters the authentication data as required by the ACS UI.

Step 14 The Browser

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

Step 15 The ACS

The ACS shall:

- Seq 3.166 **[Req 464]** If the ACS determines that Decoupled Authentication Fallback is necessary, inform the Cardholder of Decoupled Authentication in the final CRes message using the Information UI template as defined in Chapter 4.
- Seq 3.167 **[Req 123]** Check the authentication data received and assess the status of the authentication:
- If correct, then the ACS:
 - Increments the Interaction Counter
 - Sets Transaction Status = Y
 - Sets the ECI value as defined by the specific DS
 - Generates the Authentication Value as defined by the DS
 - Continues with Step 16
 - If the authentication has failed, is not completed or the Cardholder has selected to cancel the authentication, then the ACS:

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

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

Step 16 The ACS

The ACS shall for all Challenge Flow transactions (ARes Transaction Status = C) and for a Decoupled Authentication transaction (ARes Transaction Status = D) once the authentication as defined in **[Req 326].b** has completed or the timer as defined in **[Req 326].a** has expired, do the following:

- Seq 3.168 **[Req 465]** For a Challenge Flow (ARes Transaction Status = C), if 3DS Requestor Decoupled Request Indicator = F or B, and if the ACS has determined that Decoupled Authentication Fallback is necessary, set Transaction Status = D.
- Seq 3.169 **[Req 124]** Format the RReq message as defined in Table B.8.
- Seq 3.170 **[Req 125]** Establish a secure link with the DS as defined in Section 6.1.3.2.
- Seq 3.171 **[Req 126]** If the Cardholder abandons the challenge during the processing of Step 12 through Step 14, then the ACS sets the Challenge Cancelation Indicator to the appropriate value in the RReq message. Refer to Annex A for specific values.
- Seq 3.172 **[Req 127]** Send the RReq message to the DS.
- Seq 3.173 **[Req 128]** Ensure that one RReq message is sent to the DS for each ARes message with a Transaction Status = C or D.
- Seq 3.174 **[Req 347]** Ensure for a Decoupled Authentication transaction that:
 - An RReq message is sent immediately upon obtaining an authentication result (whether successful or not).
 - An RReq message without an authentication result (Transaction Status = U) is sent when the 3DS Requestor Decoupled Max Time expires—with a grace period of 1 hour.

Note: It is recommended that an RReq message with Transaction Status = U contains Transaction Status Reason = 24 or 26 and Challenge Cancelation Indicator = 03.

Step 17 The DS

The DS shall:

- Seq 3.175 **[Req 129]** Receive the RReq message from the ACS and Validate as defined in Section 5.9.8.
If the message is in error the DS **ends processing**.
- Seq 3.176 **[Req 130]** Establish a secure link with the 3DS Server as defined in Section 6.1.2.2 using the 3DS Server URL data element extracted from the AReq message and stored in **[Req 99]**.
- Seq 3.177 **[Req 131]** Send the RReq message to the 3DS Server using the secure link established in **[Req 130]**.

Step 18 The 3DS Server

The 3DS Server shall:

- Seq 3.178 **[Req 132]** Receive the RReq message or Error Message from the DS and Validate as defined in Section 5.9.9.
If the message is in error the 3DS Server **ends processing**.
- Seq 3.179 **[Req 466]** If Transaction Status = D and if 3DS Requestor Decoupled Request Indicator = F or B, set Results Message Status to 04.

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

- Seq 3.180 **[Req 133]** Format the RRes message as defined in Table B.9 and send it to the DS using the secure link established in **[Req 130]**.
- Seq 3.181 **[Req 348]** For a Decoupled Authentication transaction, at a minimum wait the specified 3DS Requestor Decoupled Max Time plus 1 hour, 30 seconds for the RReq, If an RReq message is never received, further processing is outside the scope of 3-D Secure processing.

Note: If the RReq message is not received, then the 3DS Server should assume that the Decoupled Authentication is not successful.

Note: For Payment Authentication, the Merchant can now proceed with Authorisation processing with its Acquirer. However, the Merchant may first want to receive confirmation that the Cardholder has not abandoned the transaction.

Step 19 The DS

The DS shall:

- Seq 3.182 **[Req 134]** Receive the RRes message or Error Message from the 3DS Server and Validate as defined in Section 5.9.10.
If the message is in error the DS **ends processing**.
- Seq 3.183 **[Req 135]** Log transaction information as required by the DS rules.
- Seq 3.184 **[Req 136]** Send the RRes message to the ACS through the secure link established in **[Req 125]**.

Step 20 The ACS

The ACS shall:

Seq 3.185 **[Req 137]** Receive the RRes message or Error Message from the DS and Validate as defined in Section 5.9.12.

If the RRes message is in error, the ACS **ends processing**.

Note: 3-D Secure processing completes for Decoupled Authentication transactions.

Step 21 The ACS

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

Seq 3.186 **[Req 138]** Format the final CRes message as defined in Table B.5.

Seq 3.187 **[Req 139]** Base64url-encode the final CRes message and include, if present in the CReq message, the 3DS Requestor Session Data in HTML form (as defined in Table A.3).

Seq 3.188 **[Req 140]** Send the final CRes message via an HTTP POST (for example, utilising JavaScript) through the Browser to the Notification URL that was sent in the initial AReq message using the secure link established in Step 10.

Step 22 The 3DS Requestor Environment

The 3DS Requestor Environment continues with the checkout process and takes the appropriate action.

The 3DS Requestor Environment:

- For a transaction with a Challenge, receives the CRes message at the Notification URL as defined in Section B.4.
- Conveys the appropriate response to appropriate 3DS Requestor Environment components and closes the challenge iframe.

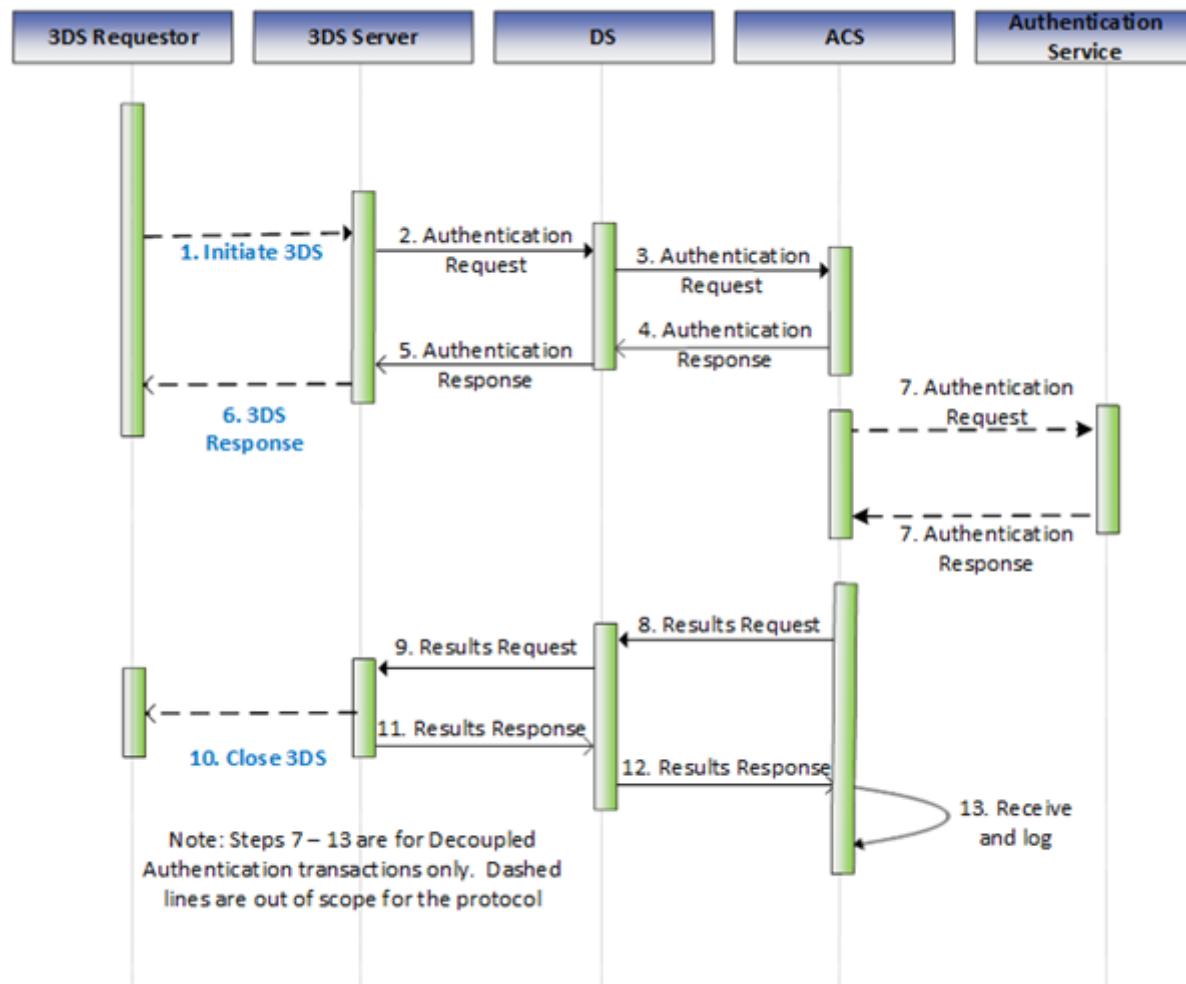
Note: The 3DS Requestor should notify their 3DS Server and DS if invalid CRes messages are being received.

Note: 3-D Secure processing completes.

3.4 3RI-based Requirements

The Steps in the 3RI flow are outlined in Figure 3.4 with detailed requirements following the figure. The 3RI flow supports two primary main use cases: confirmation of account information (accomplished through Steps 1–6) and Cardholder authentication (accomplished through Steps 1–13).

Figure 3.4: 3-D Secure Processing Flow Steps—3RI-based



The 3-D Secure processing flow for 3RI-based implementations contains the following Steps:

Step 1: The 3DS Requestor

The 3DS Requestor is responsible for gathering the information for the AReq message assembled by the 3DS Server.

As introduced in Section 2.1.1, this specification does not require a specific component to gather the information, only that the information is available for the 3DS Server when the AReq message is built. This information can include:

- Cardholder Account Information
- Merchant Risk Indicator
- 3DS Requestor Prior Transaction Authentication Information
- Payment Information
- Non-Payment Information
- Cardholder information

This information is made available to the 3DS Server.

The 3DS Server shall:

- Seq 3.189 **[Req 271]** Ensure, if the 3DS Requestor and 3DS Server are separate components, that data transferred between the components is protected at a level that satisfies Payment System security requirements with mutual authentication of both servers.
- If the communication is not established as required, the 3DS Server **ends 3-D Secure processing**.
 - If the communication channel(s) within the 3DS Requestor Environment makes it impossible to have the 3DS Server receive the information within a reasonable amount of time, then this shall be reported to the 3DS Requestor and **ends 3-D Secure processing**.

Step 2 The 3DS Server

The 3DS Server shall:

- Seq 3.190 **[Req 272]** Obtain the 3DS Requestor ID, the 3DS Server Reference Number, and conditionally the Acquirer BIN.

- Seq 3.191 **[Req 302]** Generate the 3DS Server Transaction ID.

- Seq 3.192 **[Req 273]** Ensure availability of the necessary information for the AReq message (as defined in Table B.1) gathered by components within the 3DS Requestor Environment.

If the necessary information is not available, the 3DS Server **ends 3-D Secure processing**.

- Seq 3.193 **[Req 274]** Determine which DS the authentication transaction needs to be sent based on the BIN (as defined in ISO 7812) and optionally other Cardholder account information.

- Seq 3.194 **[Req 275]** Establish a secure link with the DS as defined in Section 6.1.2.1.

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

- Seq 3.195 **[Req 423]** Use the protocol version lists from the ACS Protocol Versions and DS Protocol Versions obtained from the PRes message to set the highest common Message Version Number.

- If no PRes message information is available, then the 3DS Server may use a Message Version Number supported by the 3DS Server.
- Seq 3.196 **[Req 467]** In the case of Decoupled Authentication Fallback, the 3DS Server initiates a 3RI authentication within 60 seconds of receiving the RReq message from the previous transaction, containing:
- 3DS Requestor Decoupled Request Indicator = Y
 - 3DS Requestor Prior Transaction Authentication Information object:
 - 3DS Requestor Prior Transaction Reference = ACS Transaction ID from the RReq message indicating that Decoupled Authentication is to be performed
 - 3DS Requestor Prior Transaction Authentication Method = 02 (Cardholder challenge occurred by ACS).
- Seq 3.197 **[Req 276]** Format the AReq message as defined in Table B.1.
- Seq 3.198 **[Req 277]** Send the AReq message to the DS using the secured link established in **[Req 275]**.
- If the 3DS Server receives a failure when communicating with the DS or does not get a response (as defined in Section 5.5), then the 3DS Server **ends 3-D Secure processing**.
- Step 3 The DS**
- The DS shall:
- Seq 3.199 **[Req 278]** Receive the AReq message from the 3DS Server and Validate as defined in Section 5.9.1.
- If the message is in error the DS **ends processing**.
- Seq 3.200 **[Req 279]** Generate the DS Transaction ID.
- Seq 3.201 **[Req 280]** Check that the Message Version Number is supported by the DS and the ACS.
- If not, the DS returns to the 3DS Server EITHER:
- an ARes message (as defined in Table B.2) with Transaction Status set to the appropriate response as defined by the specific DS and **ends processing**, OR
 - an Error Message (as defined in Section A.9) with Error Component = D and Error Code = 102 and **ends processing**.
- Seq 3.202 **[Req 281]** Further Check the data elements in the AReq message as follows:
- If the 3DS Server Reference Number does not represent a participating 3DS Server, then the DS returns to the 3DS Server an Error Message (as defined in Section A.9) with Error Component = D and Error Code = 303 then **ends processing**.
 - If Merchant Category Code (MCC) is not valid for the specific DS, then the DS returns to the 3DS Server an Error Message (as defined in Section A.9) with Error Component = D and Error Code = 306 and **ends processing**.
- Seq 3.203 **[Req 282]** Determine if the Cardholder Account Number received in the AReq message is in a participating account range.

If not, the DS returns to the 3DS Server an ARes message (as described in Table B.2) with the Transaction Status set to the appropriate value as defined by the specific DS and **ends processing**.

Seq 3.204 **[Req 283]** Determine if the Cardholder Account Number is in an account range that has an ACS capable of processing 3-D Secure messages.

If not, the DS returns to the 3DS Server an ARes message (as described in Table B.2) with the Transaction Status set to the appropriate value as defined by the specific DS and **ends processing**.

Seq 3.205 **[Req 427]** Store the 3DS Server URL with the DS Transaction ID (for possible RReq message processing).

Seq 3.206 **[Req 285]** Establish a secure link with the ACS as defined in Section 6.1.3.1.

If the connection cannot be established with the ACS, then the DS proceeds as specified in **[Req 233]** and **ends processing**.

Note: The DS may maintain multiple ACS URLs. If the first URL attempted is not available, then the DS will attempt to connect to one of the alternate URLs.

The DS shall:

Seq 3.207 **[Req 286]** Send the AReq to the ACS using the secured link established in **[Req 285]**.

- If the DS does not receive an ARes message from the ACS (as defined in Section 5.5), the DS returns to the 3DS Server a message as specified in **[Req 235]** and **ends processing**.

Step 4 The ACS

The ACS shall:

Seq 3.208 **[Req 287]** Receive the AReq message from the DS and Validate as defined in Section 5.9.2.

If the message is in error the ACS **ends processing**.

Seq 3.209 **[Req 288]** Generate the ACS Transaction ID.

Seq 3.210 **[Req 289]** Use the Cardholder Account Number from the AReq message to determine whether authentication is available for the Cardholder.

If the authentication for the Cardholder Account Number is not available, the ACS returns to the 3DS Server an ARes message (as defined in Table B.2) with the Transaction Status, and the Transaction Status Reason code set to the appropriate response as defined by the specific DS and **ends processing**.

The ACS should:

Seq 3.211 **[Req 290]** Use the values of the 3RI Indicator, the 3DS Requestor Decoupled Request Indicator and the 3DS Requestor Prior Transaction Authentication Information received in the AReq message when evaluating the transaction disposition as defined in **[Req 291]**.

The ACS shall:

Seq 3.212 **[Req 291]** Evaluate the values received in the AReq message and determine whether the 3RI transaction⁵ is:

⁵ The decisioning process for this action is outside the scope of this specification.

- authenticated (Transaction Status = Y)
- requiring a Cardholder challenge using Decoupled Authentication (Transaction Status = D)
- not authenticated (Transaction Status = N)
- not authenticated, but a proof of authentication attempt (Authentication Value) was generated (Transaction Status = A)⁶
- not authenticated because authentication could not be performed due to a technical or other problem (Transaction Status = U)
- not authenticated because the Issuer is rejecting authentication and requesting that authorisation not be attempted (Transaction Status = R)
- data sent for informational purposes only, an authentication not requested by the 3DS Server (Transaction Status = I)

Seq 3.213 **[Req 292]** If a transaction is deemed authenticated (Transaction Status is = Y or A) then the ACS performs the following:

- For a Payment Authentication (Message Category = 01-PA), the ACS shall:
 - Generate the ECI value and Authentication Value and include in the ARes message as defined by the specific Payment System.
- For a Non-Payment Authentication (Message Category = 02-NPA), the ACS *may*:
 - Generate the ECI value and Authentication Value and include in the ARes message as defined by the specific Payment System.
 - Assign an appropriate Transaction Status Reason value as defined by the specific DS and include in the ARes message.
- Whether the ECI Indicator/Authentication Value and/or the Transaction Status Reason value is included in the ARes message is defined by the specific DS.

Seq 3.214 **[Req 328]** If a Decoupled Authentication challenge is deemed necessary (Transaction Status = D), the ACS determines whether an acceptable challenge method is supported by the ACS based in part on the following data element received in the AReq message: 3DS Requestor Decoupled Max Time. The ACS performs the following:

- a. Sets the Transaction Status = D (Decoupled Authentication).
- b. Sets the ACS Decoupled Confirmation Indicator = Y.
- c. Stores the 3DS Server Transaction ID and DS Transaction ID (for subsequent RReq message processing).

Seq 3.215 **[Req 293]** Complete formatting of the ARes message as defined in Table B.2.

Seq 3.216 **[Req 294]** Send the ARes message to the DS using the secure link established in **[Req 285]**.

Step 5 The DS

The DS shall:

⁶ Support for attempts is determined by each DS.

- Seq 3.217 **[Req 308]** Check the data elements in the ARes message as follows.
- If the ACS Reference Number does not represent a participating ACS then the DS shall:
 - Return to the 3DS Server an Error Message (as defined in Section A.9) with Error Component = D and Error Code = 303 and **ends processing**, OR
 - Send an ARes message to the 3DS Server (as defined in Table B.2) with Transaction Status set to the appropriate response as defined by the specific DS.
- Seq 3.218 **[Req 295]** Receive the ARes message or Error message from the ACS and Validate as defined in Section 5.9.3.
If the message is in error, the DS **ends processing**.
- Seq 3.219 **[Req 296]** Log the transaction information as required by the DS rules.
- Seq 3.220 **[Req 297]** Send the ARes message to the 3DS Server received from the ACS using the secure link established in **[Req 275]**.
- Seq 3.221 **[Req 412]** If the DS creates the ARes message on the ACS's behalf (for example, the DS returns a Transaction Status = A), then the DS sets the ACS Reference Number equal to the DS Reference Number and the ACS Transaction ID equal to the DS Transaction ID.

Step 6 The 3DS Server

The 3DS Server shall:

- Seq 3.222 **[Req 298]** Receive the ARes message or Error Message from the DS and Validate as defined in Section 5.9.4.
If the message is in error the 3DS Server **ends processing**.
- Seq 3.223 **[Req 299]** Send necessary information (as defined in Table B.2) from the ARes message to the 3DS Requestor Environment.
- Seq 3.224 **[Req 357]** If the Cardholder Information Text has been provided by the ACS for this transaction the 3DS Server shall ensure the Cardholder Information Text is conveyed on the 3DS Requestor website.

Note: 3-D Secure processing completes for non-Decoupled Authentication transactions.

Note: For a Decoupled Authentication transaction, as defined in **[Req 353], the 3DS Server is now expected to wait for the ACS to authenticate the Cardholder at which point in time the ACS will send an RReq message.**

Step 7 The ACS

The ACS shall:

- Seq 3.225 **[Req 330]** For a Decoupled Authentication transaction (Transaction Status = D), do the following:
- a. Start a timer against the 3DS Requestor Decoupled Max Time
 - b. Authenticate the Cardholder. How an authentication decision is made is outside the scope of this specification. However, the ACS's objective is to complete the Cardholder authentication before the 3DS Requestor Decoupled Max Time expires.

Step 8 The ACS

The ACS shall for a Decoupled Authentication transaction (initial Transaction Status = D) once the authentication as defined in **[Req 330].b** has completed, or the timer as defined in **[Req 330].a** has expired, do the following:

- Seq 3.226 **[Req 349]** Format the RReq message as defined in Table B.8.
- Seq 3.227 **[Req 350]** Establish a secure link with the DS as defined in Section 6.1.3.2.
- Seq 3.228 **[Req 351]** Send the RReq message to the DS.
- Seq 3.229 **[Req 352]** Ensure that one RReq message is sent to the DS for each ARes message with Transaction Status = D.
- Seq 3.230 **[Req 353]** Ensure that:
- An RReq message is sent immediately upon obtaining an authentication result (whether successful or not).
 - An RReq message without an authentication result (Transaction Status = U) is sent when the 3DS Requestor Decoupled Max Time expires—with a grace period of 1 hour.

Note: It is recommended that an RReq message with Transaction Status = U contains Transaction Status Reason = 24 or 26 and Challenge Cancelation Indicator = 03.

Step 9 The DS

The DS shall:

- Seq 3.231 **[Req 332]** Receive the RReq message from the ACS and Validate as defined in Section 5.9.8.
If the message is in error the DS **ends processing**.
- Seq 3.232 **[Req 333]** Establish a secure link with the 3DS Server as defined in section 6.1.2.2 using the 3DS Server URL extracted from the AReq message.
- Seq 3.233 **[Req 334]** Send the RReq message to the 3DS Server using the secure link established in **[Req 333]**.

Step 10 and Step 11 The 3DS Server

The 3DS Server shall:

- Seq 3.234 **[Req 335]** Receive the RReq message or Error Message from the DS and Validate as defined in Section 5.9.9.
If the message is in error, the 3DS Server **ends processing**.
- Seq 3.235 **[Req 336]** Format the RRes message as defined in Table B.9 and send to the DS using the secure link established in **[Req 333]**.
- Seq 3.236 **[Req 354]** For a Decoupled Authentication transaction, at a minimum wait the specified 3DS Requestor Decoupled Max Time plus 1 hour, 30 seconds for the RReq message. If an RReq message is never received, further processing is outside the scope of 3-D Secure processing.

Note: If the RReq message is not received, then the 3DS Server should assume that the Decoupled Authentication is not successful.

- Seq 3.237 **[Req 337]** Convey the appropriate response to the 3DS Requestor.

Step 12 The DS

The DS shall:

- Seq 3.238 **[Req 338]** Receive the RRes message or Error Message from the 3DS Server and Validate as defined in Section 5.9.10.
If the message is in error the DS **ends processing**.
- Seq 3.239 **[Req 339]** Log the transaction information as required by the DS.
- Seq 3.240 **[Req 340]** Send the RRes message to the ACS as received from the 3DS Server using the secure link established in **[Req 350]**.

Step 13 The ACS

The ACS shall:

- Seq 3.241 **[Req 341]** Receive and log the RRes message or Error Message from the DS and Validate as defined in Section 5.9.11.
If the message is in error the ACS **ends processing**.

Note: 3-D Secure processing completes.

3.5 SPC-based Authentication Requirements

SPC (Secure Payment Confirmation) provides a method to perform a challenge using pre-established FIDO credentials when using a Browser. The SPC authentication can be initiated by the 3DS Requestor via an extra AReq/ARes message pair or by the ACS via a standard Browser Challenge Flow.

This section outlines the differences to a normal Browser flow (Section 3.3) for the two variants of using SPC as a challenge method.

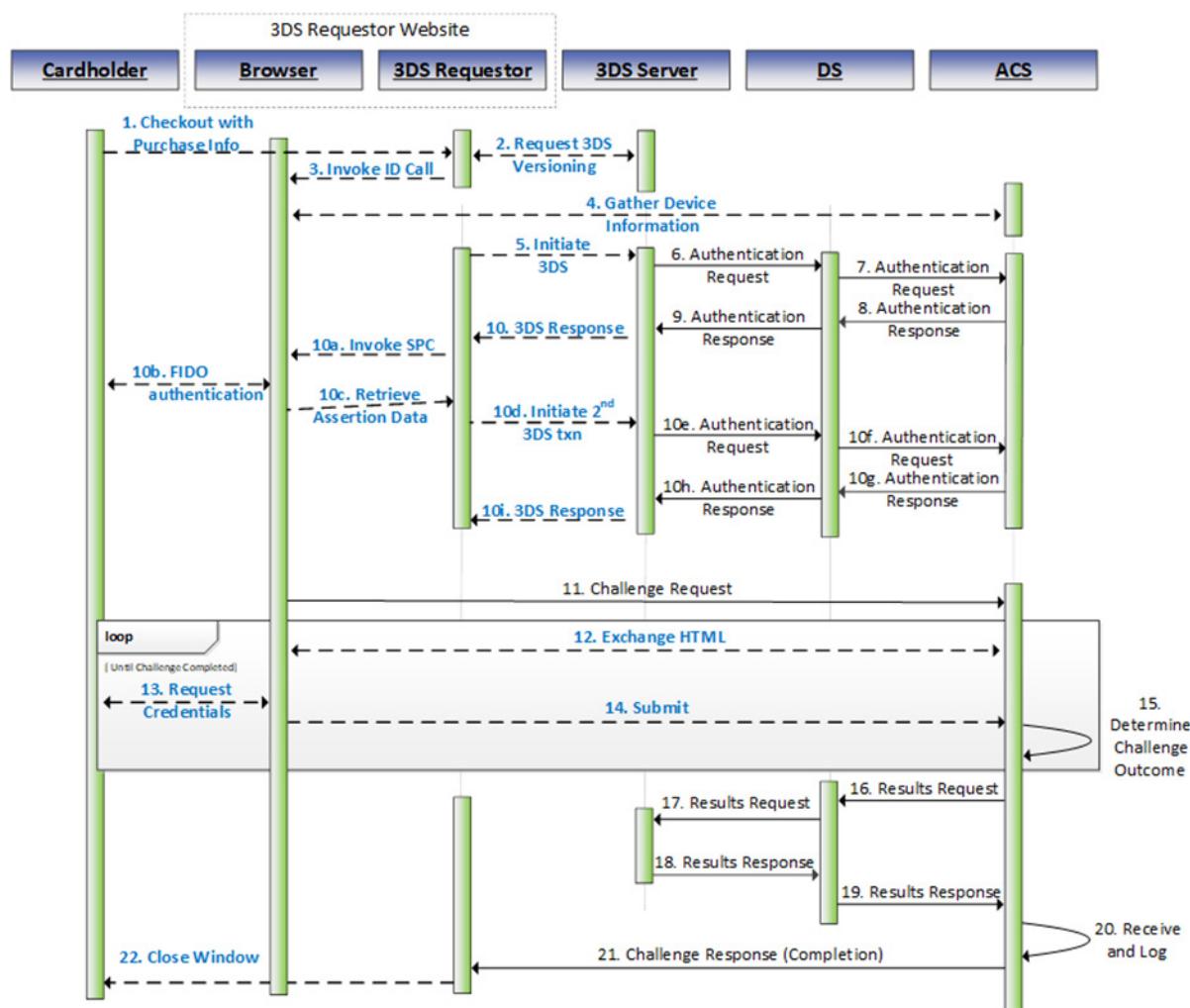
For an SPC authentication to execute correctly, the following prerequisites apply:

1. The ACS has an enrolled FIDO authenticator on the device for this Cardholder.
2. The 3DS Requestor and/or the ACS have detected that the Cardholder Browser supports the related SPC APIs (`allow="payment *; publickey-credentials-get *`). For the ACS, this information can be obtained via the Browser User Agent data element or via data obtained via the 3DS Method.

3.5.1 3DS Requestor Initiates SPC Authentication

The steps for an authentication flow with SPC initiated by the 3DS Requestor follow a standard Browser flow (as outlined in Figure 3.3), with nine additional steps (Steps 10a–10i in Figure 3.5). The full flow, including the steps not affected by SPC Authentication is outlined in Figure 3.5.

Figure 3.5: 3-D Secure Processing Flow Steps—SPC-based



The 3DS Requestor-initiated SPC-based authentication flow is identical to a standard 3-D Secure processing flow as defined in Section 3.3 with the following exceptions and additions:
Step 5: The 3DS Requestor initiates a 3DS authentication indicating to the 3DS Server that it can support SPC authentication for this transaction and whether the Browser supports the SPC API.

Seq 3.242 **[Req 447]** The 3DS Requestor website shall display the Processing screen as per the requirements in Section 4.3.1.1 until the SPC API is invoked in Step 10a.

Step 6: The 3DS Server recognises that the ACS supports SPC-based authentication in the ACS Information Indicator.

Seq 3.243 **[Req 448]** The 3DS Server shall set the 3DS Requestor SPC Support = Y in the AReq message.

Step 8: The ACS recognises that SPC-based authentication is supported.

Seq 3.244 **[Req 449]** If the ACS determines that SPC is the selected authentication method, the ACS shall return the following:

- Transaction Status = S

- List of enrolled FIDO credentials (WebAuthn Credential list) associated with this Cardholder
- SPC Transaction Data (See Table A.28)

Note: Depending on implementation, it is possible that the DS performs the functions described in Step 8 on behalf of the ACS. In such cases, the data are provided by the DS to the ACS in Step 7 and to the 3DS Server in Step 9.

Step 10 The 3DS Server

The 3DS Server shall:

Seq 3.245 **[Req 443]** If Transaction Status = S (SPC authentication performed by the 3DS Requestor):

- a. Send necessary information from the ARes message (as defined in Table B.2) to the 3DS Requestor Environment, in particular the WebAuthn Credential list, and the SPC Transaction Data.
- b. Continue with **Step 10a**.

Step 10a The 3DS Requestor

The 3DS Requestor invokes the SPC authentication (SPC API) against the WebAuthn Credential list returned in the ARes message and provides the SPC Transaction Data.

Seq 3.246 **[Req 450]** The 3DS Requestor shall not change or store any of the data received for the SPC authentication from the 3DS Server.

Step 10b The Cardholder

The Cardholder authenticates using the FIDO authenticator on his/her device.

Step 10c The 3DS Requestor

The 3DS Requestor retrieves the Assertion Data from the SPC API call.

Step 10d The 3DS Requestor

The 3DS Requestor initiates a second 3DS Authentication Request to return Assertion Data to the ACS, and displays the processing screen as defined in Section 4.3.1.1 during the AReq message processing.

Step 10e The 3DS Server

The 3DS Server sends a second AReq message with the FIDO Assertion Data.

In this step, in addition to performing the requirements defined in Step 6 of the Browser flow, the 3DS Server shall:

Seq 3.247 **[Req 444]** Include the following in the AReq message:

- a. A fresh 3DS Server Transaction ID
- b. The Assertion Data generated by the FIDO authenticator included in the 3DS Requestor Authentication Data using a 3DS Requestor Authentication Method = 09
- c. The 3DS Requestor Prior Transaction Authentication Information object:
 - o 3DS Requestor Prior Transaction Reference = ACS Transaction ID from the ARes message indicating that SPC authentication is to be performed
 - o 3DS Requestor Prior Transaction Authentication Method = 05 (SPC authentication)

- 3DS Requestor Prior Transaction Authentication Timestamp

Note: The data gathered as part of executing [Req 87] and [Req 88] in Step 6 of the Browser-based flow is identical to the information from the first AReq message.

Step 10f The DS

The DS performs the requirements as defined in Step 7 of the Browser-based flow.

The DS can connect the information in this AReq message to the information in the previous AReq message via the previous ACS Transaction ID received in the 3DS Requestor Prior Transaction Reference.

Seq 3.248 **[Req 451]** If the DS evaluates the Assertion Data on behalf of the ACS, the DS shall include the verification result in the 3DS Requestor Authentication Method Verification Indicator.

Step 10g The ACS

The ACS performs the requirements in Step 8 of the Browser-based flow and, as part of performing that step, evaluates the Assertion Data received in the AReq message.

Depending on:

- the verification of the signature in the Assertion Data;
- the consistency of the transaction data between the first and second AReq message; AND
- the consistency of the Assertion Data with the data from the AReq message,

the ACS determines the disposition of the transaction, as defined in **[Req 107]** (e.g. authenticated [Transaction Status = Y] or to be further challenged [Transaction Status = C]).

The ACS can connect the information in this AReq message to the information in the previous AReq message via the previous ACS Transaction ID received in the 3DS Requestor Prior Transaction Reference.

Step 10h The DS

The DS performs the requirements as defined in Step 9 of the Browser flow.

Step 10i The 3DS Server

The 3DS Server performs the requirements as defined in Step 10 of the Browser flow.

Note: It is expected that, if the Assertion Data is verified correctly, no further challenge is needed and that the 3DS Server will then receive an ARes message with Transaction Status = Y, thus no further 3DS processing is necessary. Step 11 through Step 22 will be executed only if the transaction disposition decision in Step 10g, results in a Transaction Status = C.

3.5.2 The ACS Initiates SPC Authentication

When the ACS initiates and performs the SPC authentication as part of a challenge, the steps are identical to a standard Browser flow (see Figure 3.3) where SPC authentication is utilised instead of the other 3DS challenge methods. This section will outline some of the details and values for specific steps when an ACS performs an SPC authentication as part of the Challenge Flow. The steps outlined is referenced via Figure 3.3. If a step in Figure 3.3 is not referenced, there is no specific SPC authentication functionality for this step.

Step 8: In this step, the ACS recognises a pre-registered FIDO authenticator on the device for this Cardholder and selects Authentication Method = 14 (SPC), in combination with

Transaction Status = C to indicate to the 3DS Server that the ACS intends to perform SPC authentication as the challenge method.

Step 12: In this step, the ACS invokes the SPC authentication (SPC API) against the Credentials registered for this Cardholder and this device using for example, a JavaScript.

Step 13: In this step, the Cardholder authenticates using the FIDO authenticator on their device (e.g., using Windows Hello, or Apple TouchID).

Step 14: In this step, the ACS retrieves the Assertion Data from the SPC authentication Browser API, using for example, a JavaScript.

Step 15: In this step, the ACS determines the challenge outcome via evaluating the Assertion Data and verifying the signature in the Assertion Data. If verified correctly, the ACS can consider the challenge verified and continue with a Transaction Status = Y.

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

This chapter provides requirements, template examples, and guidelines for building the User Interface (UI) to support 3-D Secure authentication.

4.1 3-D Secure User Interface Templates

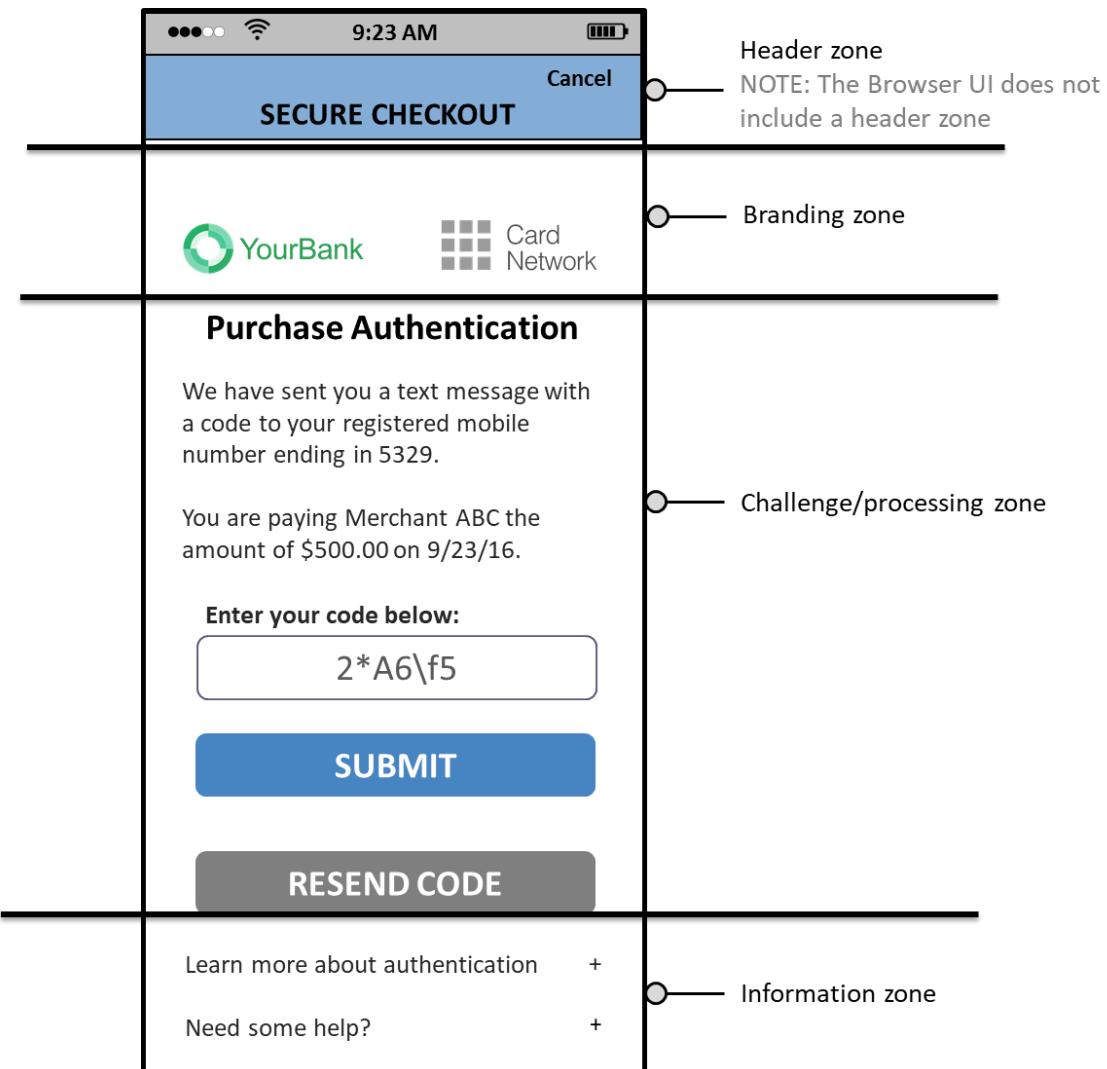
3-D Secure UI Templates provide a consistent user experience whether App-based (Native or HTML) or Browser-based. Issuers will work with their ACS to determine their specific UI content, and the UI format is driven by the Templates.

To facilitate this consistency, the UI layout is defined in zones as follows:

- **Header zone (Zone 1)**—Contains all labels managed by the 3DS Requestor and is located at the top of the screen.
- **Branding zone (Zone 2)**—Contains all logos and is located between the Header and Challenge zone.
- **Challenge/Processing zone (Zone 3)**—Contains processing and challenge information and is located between the Branding zone and the Information zone.
- **Information zone (Zone 4)**—Contains additional information for the cardholder and is located at the bottom of the screen.

Figure 4.1 illustrates the zones and placement of UI data elements within the zones.

Figure 4.1: UI Template Zones—Portrait

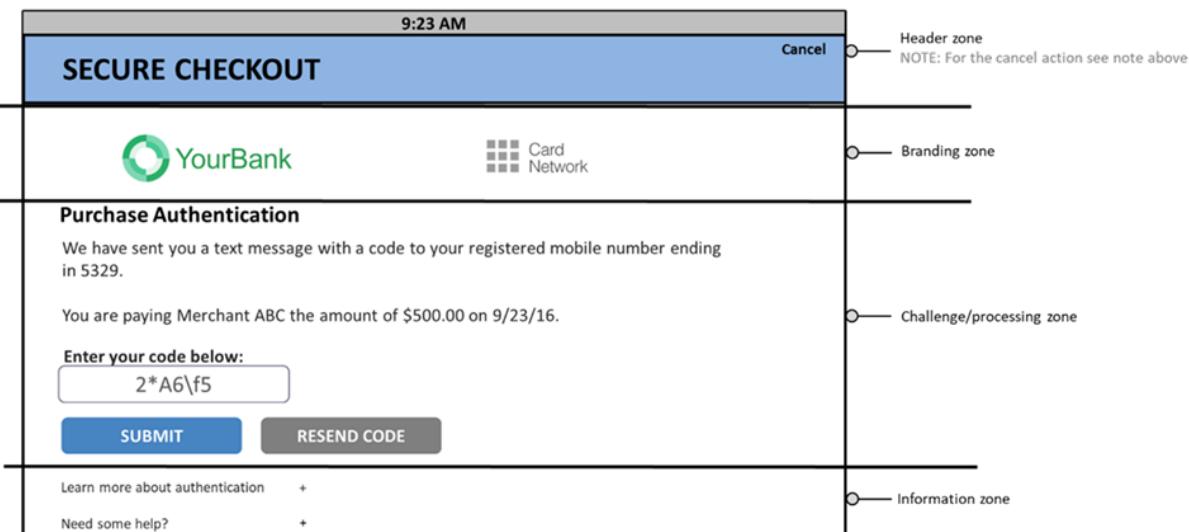


Note: The landscape UI layout is defined to provide a consistent user experience across all consumer devices or ecosystems. For example:

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

Figure 4.2 illustrates the zones and placement of UI data elements within the zones in landscape mode.

Figure 4.2: UI Template Zones—Landscape



Note: The Cancel action can be implemented as a function on a controller for the platform.

Note: Activation During Shopping (ADS) is not a supported UI template within the 3-D Secure specification.

Note: The user interface provided by the ACS for Decoupled Authentication is outside the scope of the EMV 3-D Secure protocol.

Note: Cardholder Terms and Conditions is not supported within 3-D Secure UI templates unless regulatory requirements mandate such inclusion during cardholder payment authentication.

The 3DS SDK shall:

- Seq 4.1 **[Req 314]** Support all Device Rendering Options.
- Seq 4.2 **[Req 395]** Support the UI template orientation(s) (i.e., portrait and landscape) according to the App Screen Orientation.
- Seq 4.3 **[Req 358]** For the Native UI Type, display UI data elements provided by the ACS within the applicable zones as defined in Table A.20 and depicted in Figure 4.1 and Figure 4.2. The expected format is depicted in sections 4.2.3 and 4.2.6.
- Seq 4.4 **[Req 418]** Support full-screen vertical and horizontal scrolling for HTML UI, and at minimum, support full-screen vertical scrolling for the Native UI, for all ACS-provided content. The Header zone may remain anchored at the top of the display.
- Seq 4.5 **[Req 391]** Ensure that the Header zone for the UI does not occupy more than 10 percent of the screen height.

The ACS shall:

- Seq 4.6 **[Req 342]** Support all ACS Rendering Types for the ACS supported authentication methods, at a minimum at least one ACS UI Template for each ACS Interface.
- Seq 4.7 **[Req 359]** For the App-based HTML UI Type and Browser-based UI, create HTML with form elements within the applicable zones as outlined in Figure 4.1 and Figure 4.2. The format is outlined in sections 4.2.6 and 4.3.3.

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

Figure 4.3: UI Template Examples—All Device Channels

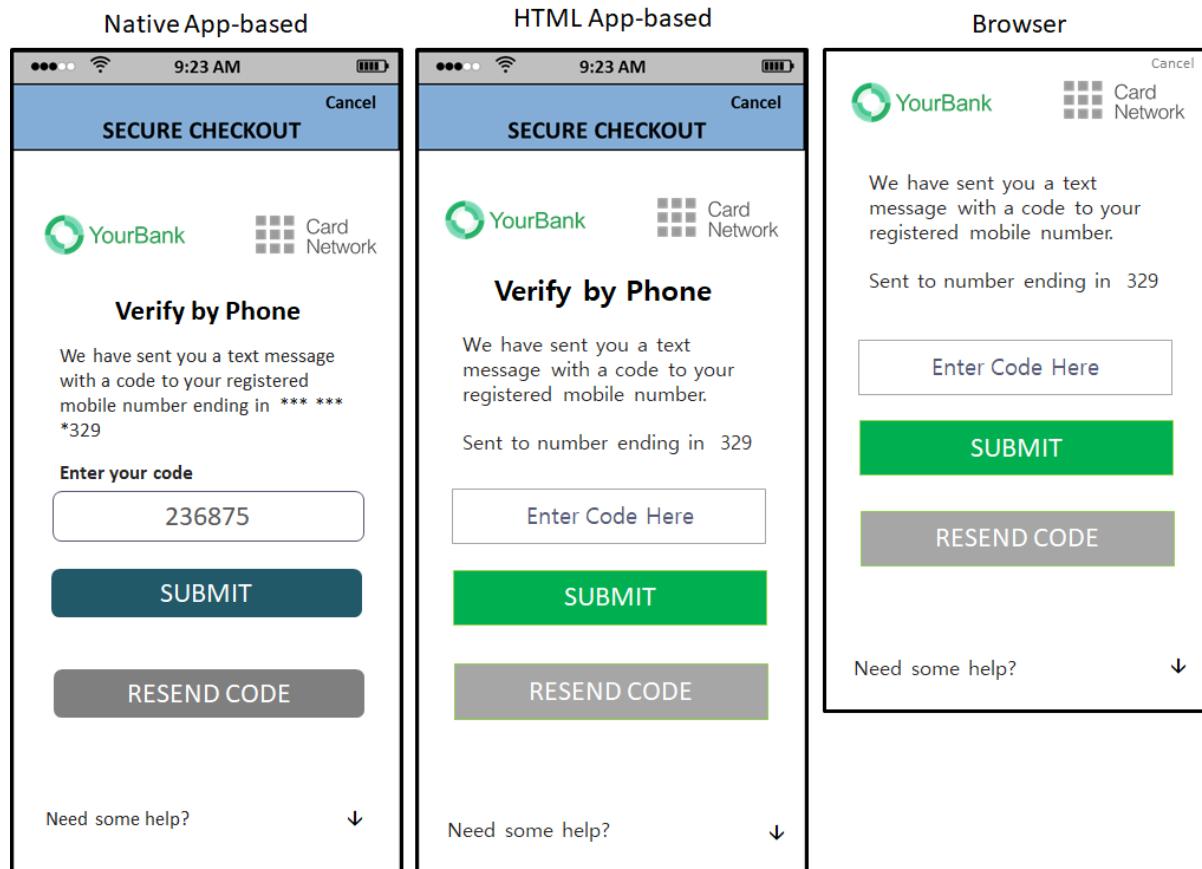
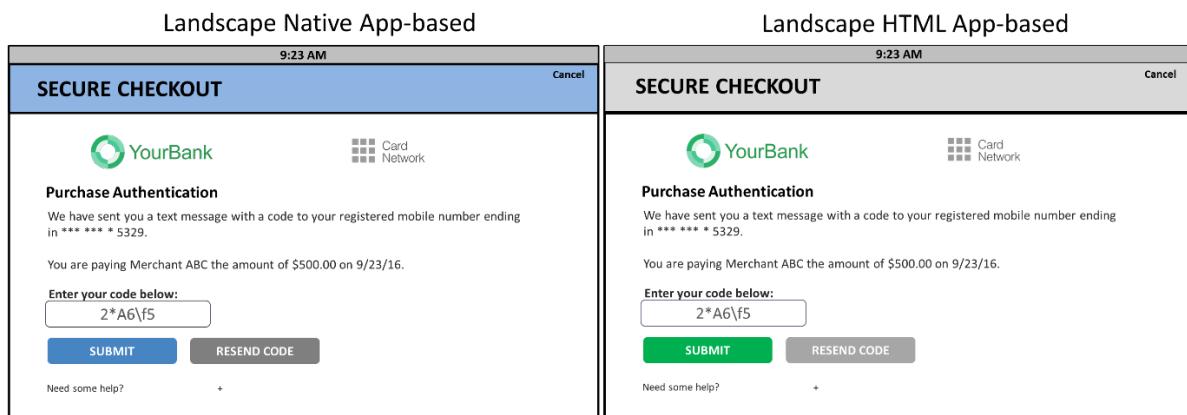


Figure 4.4 illustrates the consistency of the UI in landscape mode.

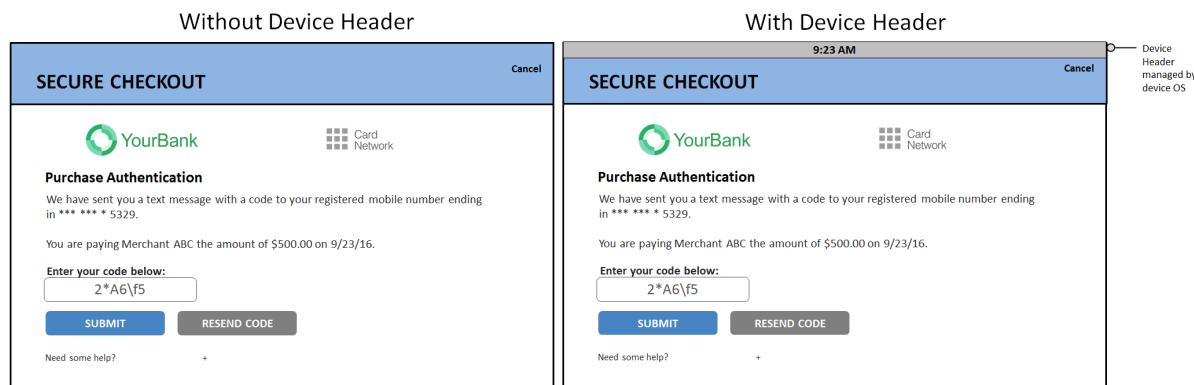
Figure 4.4: UI Template Examples—App-based—Landscape



Note: The device header is optional and may not be present depending on the 3DS Requestor implementation and OS constraints.

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

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



4.2 App-based User Interface Overview

In an App-based implementation, the 3DS Requestor App and the 3DS SDK control the rendering of the UI. Each 3DS SDK is responsible for creating the UI elements that are specific to that particular environment, for example, the operating systems (OS) on a Consumer Device. The Consumer Device determines the UI format. Either:

- Native
- HTML
- Both Native and HTML

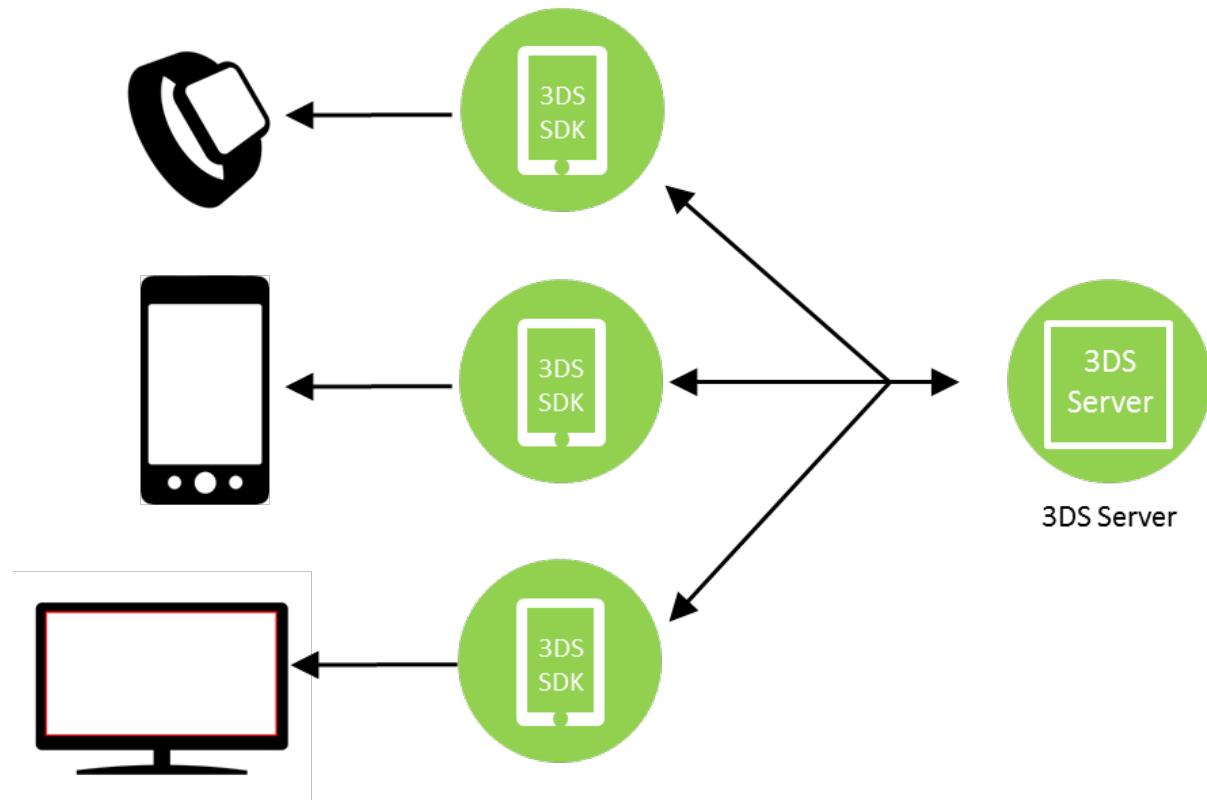
Note: The UI format should remain consistent during the CReq/CRes message exchange. For example, if starting with a Native format, then every exchange should remain a Native format.

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

Note: Some platforms may not natively support all image types.

Figure 4.6 depicts three App-based interface scenarios and illustrates a unique 3DS SDK for each.

Figure 4.6: 3DS SDK Options



4.2.1 Processing Screen Requirements

During message exchanges, a screen should be displayed to the Cardholder to indicate that 3-D Secure processing is occurring. The processing screen is applicable to both Native and HTML implementations.

Figure 4.7 and Figure 4.8 provide sample formats for the App-based Processing screen that contains both the Processing Graphic and the Logo.

Figure 4.7: Sample App-based Processing Screen—Portrait

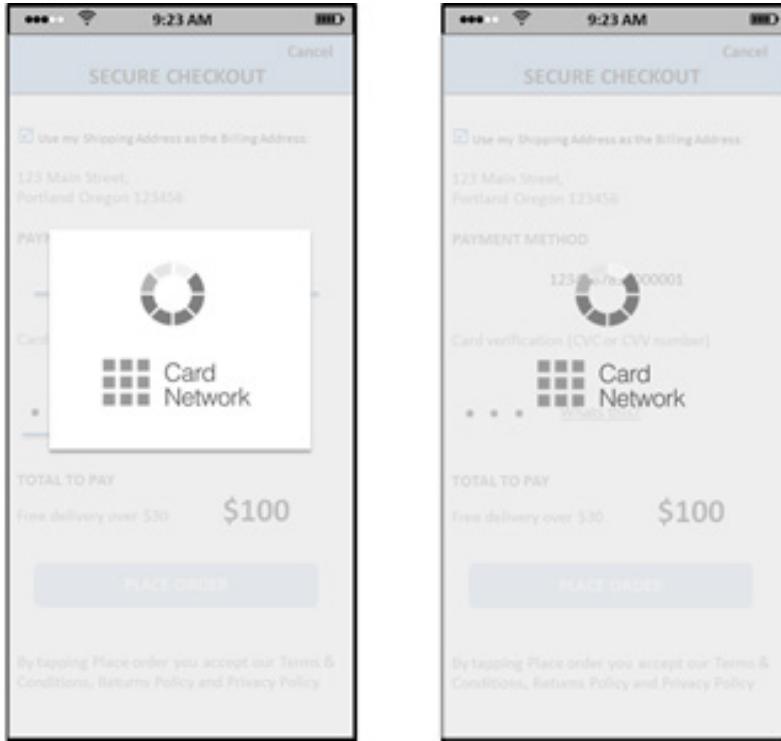
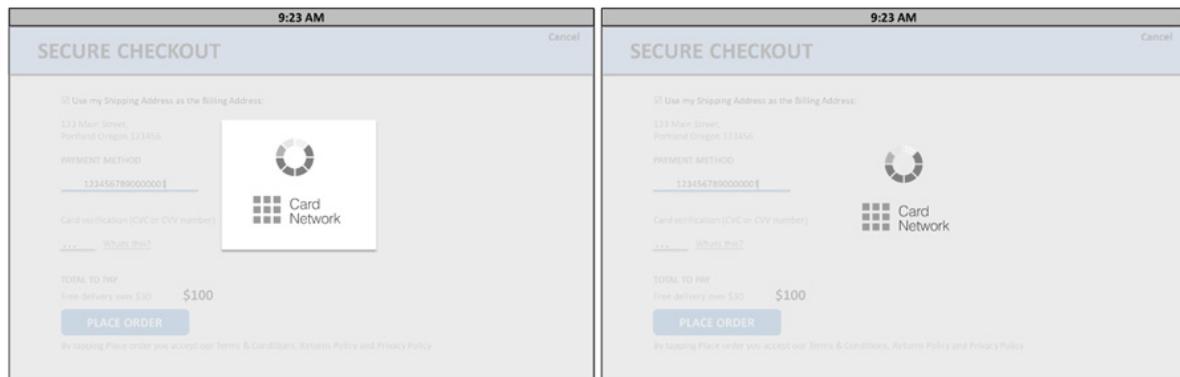


Figure 4.8: Sample App-based Processing Screen—Landscape



4.2.1.1 3DS SDK/3DS Requestor App

The 3DS SDK shall for the AReq/ARes message exchange:

- Seq 4.8 **[Req 141]** Create a Processing screen for display during AReq/ARes message cycle.
- Seq 4.9 **[Req 142]** Not include any other design element or text in the Processing screen.
- Seq 4.10 **[Req 143]** Integrate the Processing Graphic and if requested, the DS logo into the centre of the Processing screen as depicted in Figure 4.7 and Figure 4.8 with or without a white box.

Seq 4.11 **[Req 144]** Store the DS logo.

The 3DS Requestor App shall for the AReq/ARes message exchange:

Seq 4.12 **[Req 145]** Display the Processing screen supplied by the 3DS SDK during the entire AReq/ARes message cycle and overlay on the merchant checkout page as depicted in Figure 4.7 and Figure 4.8.

Seq 4.13 **[Req 146]** Display the Processing screen for a minimum of two seconds.

The 3DS Requestor App shall in case of challenge:

Seq 4.14 **[Req 388]** Set the Header zone text and the Cancel action name to be displayed by the 3DS SDK.

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

Seq 4.15 **[Req 147]** Create the Processing screen with only the default Processing Graphic (for example, a progress bar or a spinning wheel) of the Consumer Device OS without words, text or white box (See Figure 4.9–Figure 4.12).

Seq 4.16 **[Req 148]** Not include the DS logo or any other design element in the Processing screen.

Seq 4.17 **[Req 151]** Display the Processing screen for a minimum of one second during the second and subsequent CReq/CRes message cycle. (For the first CReq/CRes message cycle, see **[Req 153]**).

Seq 4.18 **[Req 389]** Ensure that the Cancel action is not actionable while displaying the Processing screen.

Figure 4.9 provides a sample format for the App-based processing flow.

Figure 4.9: Sample OTP/Text Template—App-based Processing Flow

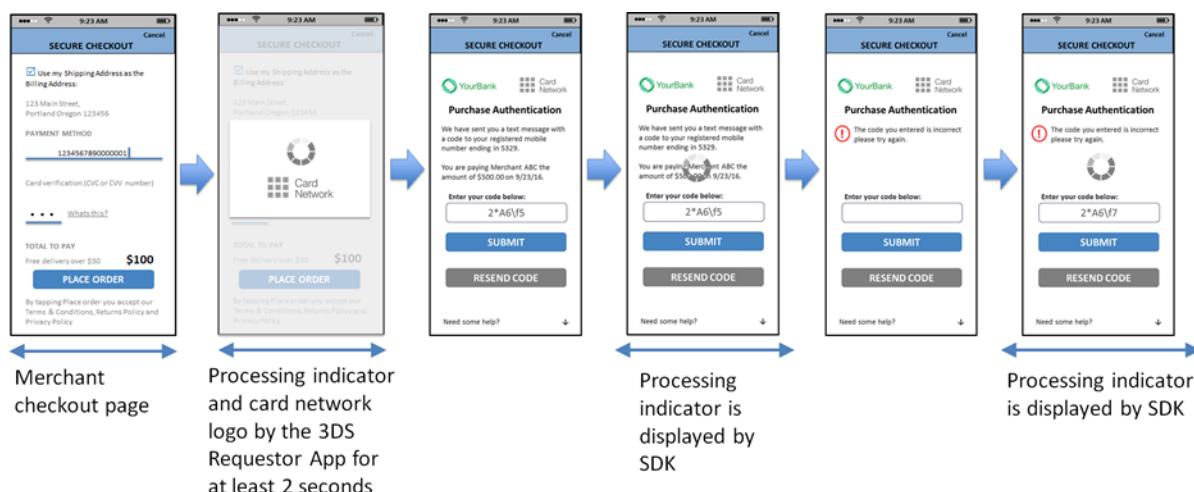


Figure 4.10 provides a sample format for the OOB template for a manual transfer to and from the OOB Authentication App for an App-based processing flow.

Figure 4.10: Sample OOB Template (Manual transfer to and from the OOB App)—App-based Processing Flow

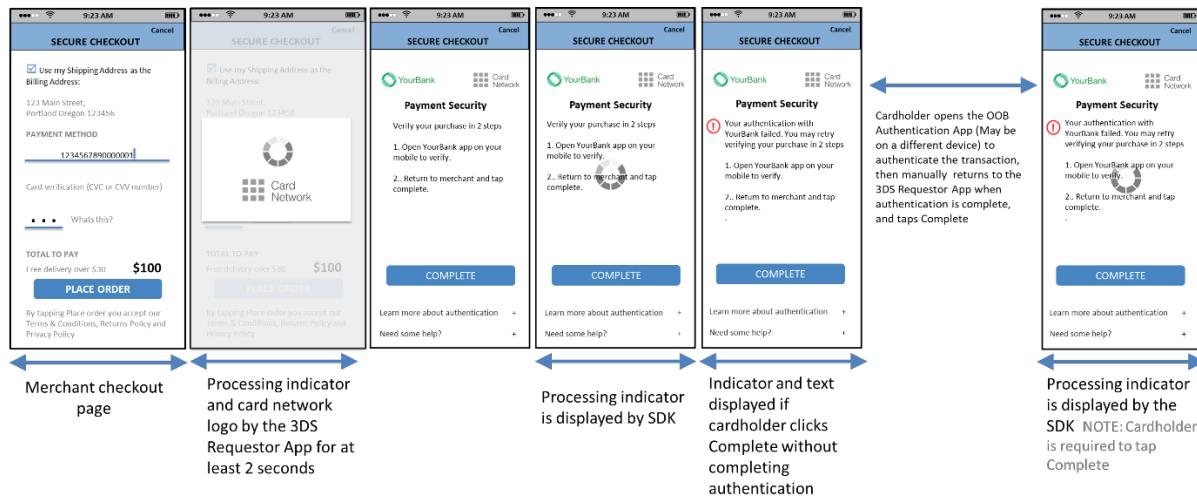


Figure 4.11 provides a sample format for the OOB template for a manual transfer to the OOB App and an automatic return to the 3DS Requestor App for an App-based processing flow. The 3DS Requestor and OOB Apps are on the same device.

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

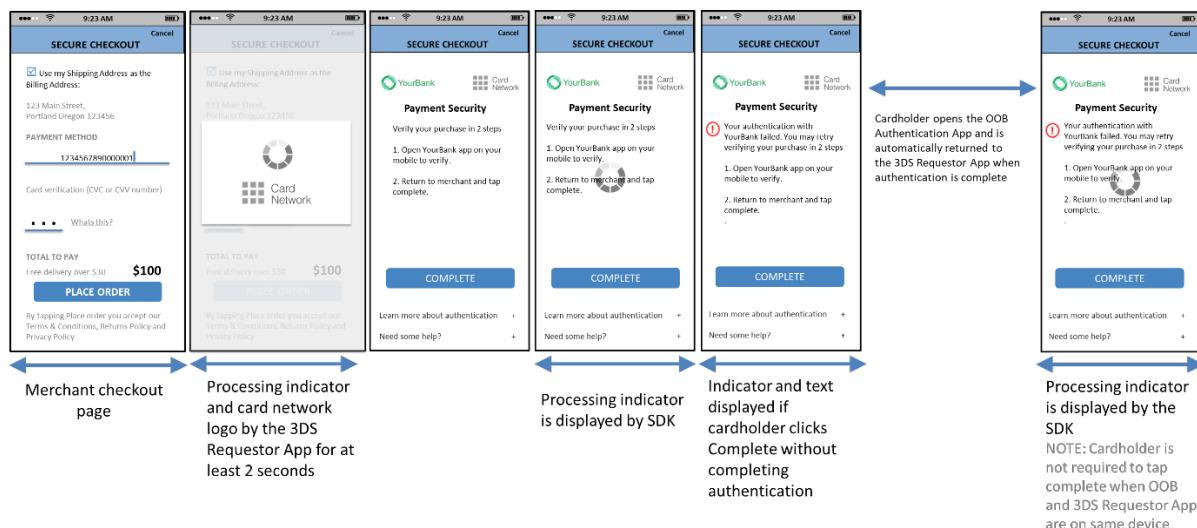


Figure 4.12 provides a sample format for the OOB template for an automatic transfer to and from the OOB App for the App-based processing flow. The 3DS Requestor and OOB Apps are on the same device.

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

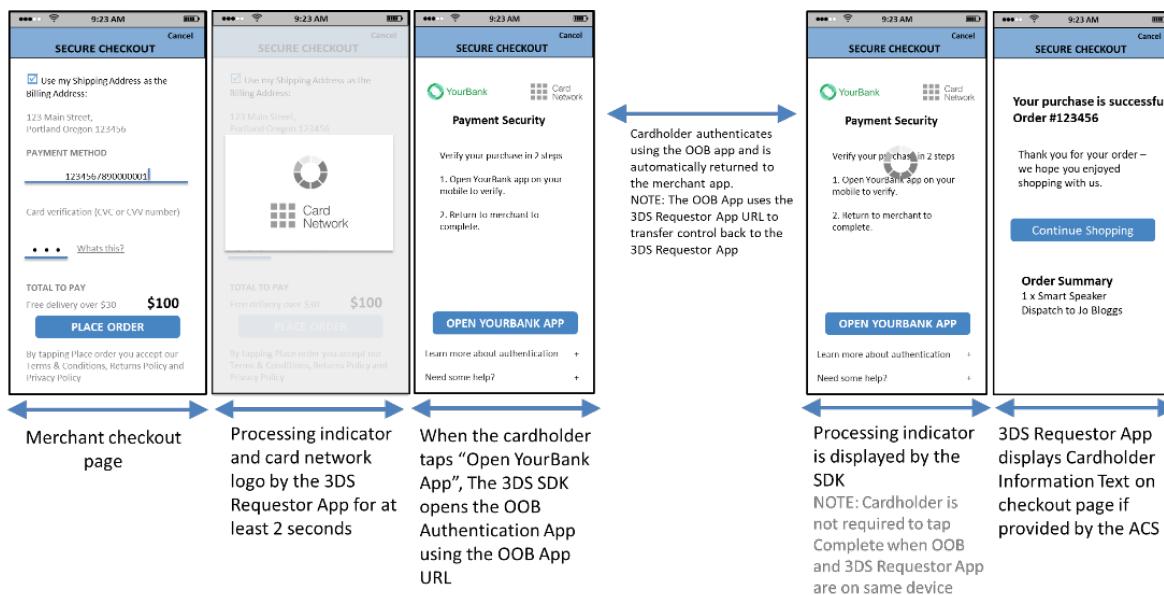
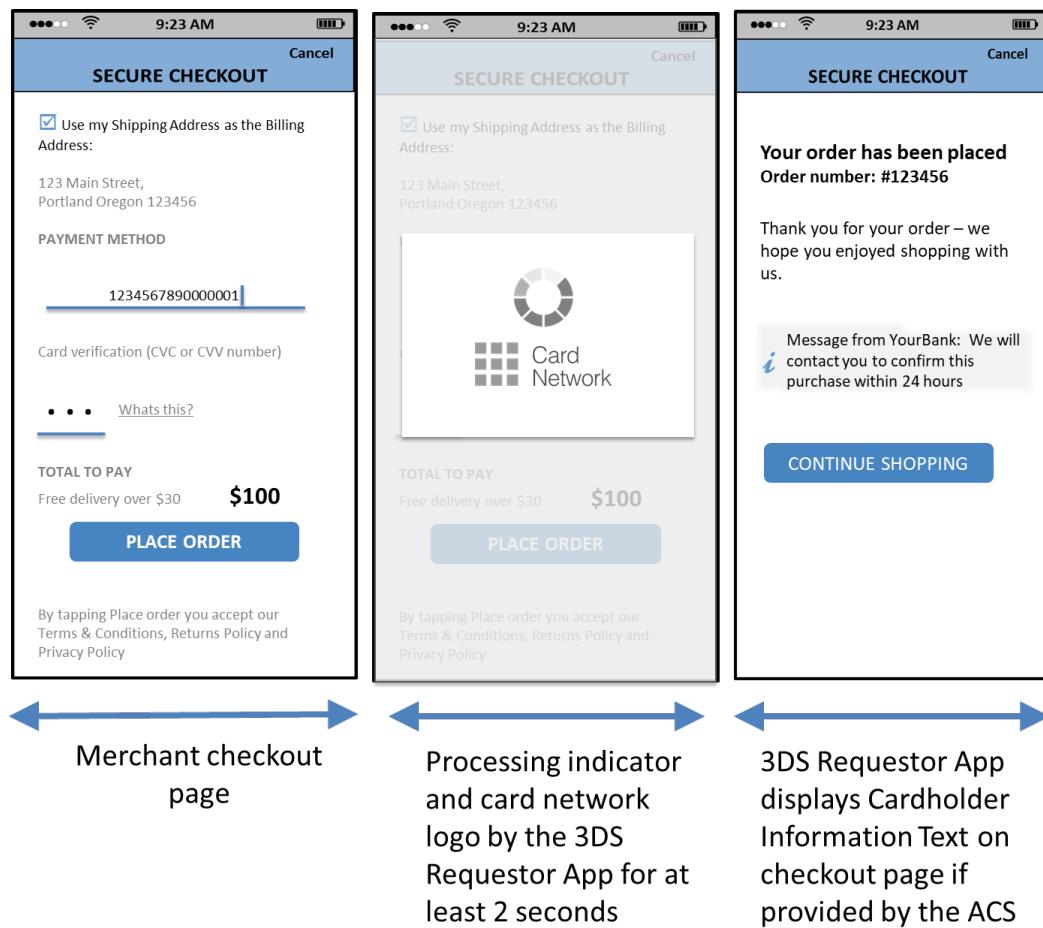


Figure 4.13 provides a sample format for the Decoupled Authentication Flow.

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



Notes:

- **There is no difference in Native vs. HTML experience.**
- **The 3DS SDK could dim the challenge window during spinning wheel display or the wheel could be translucent.**
- **The spinning wheel would be displayed through the timeout/retry process.**
- **If the Cardholder has been authenticated, then the Cardholder is returned to the merchant. If the Cardholder has not authenticated, then the UI is updated to reinforce the expected Cardholder action.**
- **For Decoupled Authentication, the UI provided by the ACS for cardholder authentication is out of scope of the EMV 3-D Secure specifications.**

4.2.2 Native UI Display Requirements

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

This format also allows for Issuer and Payment System branding. Both the 3DS Requestor App and the 3DS SDK control the rendering of the UI such that the authentication pages inherit the 3DS Requestor's UI design elements (for example, fonts). Details of the UI

rendering process through a 3DS SDK are separately described in the *EMV 3-D Secure SDK—Technical Guide*.

While the Issuer provides the content for the UI, the 3DS SDK is responsible for rendering the content. As such, the 3DS SDK can fine tune the UI to best display on the Consumer Device. With the 3DS SDK's knowledge of the device screen size and orientation, font size, etc., the 3DS SDK can optimise the content provided by the issuer (for example, by removing an extra line feed that would cause scrolling). Issuers should understand that the formatting provided in the CRes message may not be exactly what is displayed to the Cardholder.

4.2.2.1 3DS SDK/ACS

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

- Seq 4.19 **[Req 362]** For the ACS UI Type and the App Screen Orientation, display all the provided UI data elements in their applicable zones and order as defined in Table A.20 and depicted in Figure 4.1 and Figure 4.2. The expected format is depicted in sections 4.2.3 and 4.2.6.
If the 3DS SDK receives an unsupported UI data element(s) for this ACS UI Type, the 3DS SDK does not display the UI data elements, proceeds with the challenge and does not send an error message to the ACS.
- Seq 4.20 **[Req 398]** For the ACS UI Type, the 3DS SDK returns to the ACS an Error Message (as defined on Section A.9) with Error Component = C and Error Code = 201 if any mandatory UI data elements are missing as defined in Table A.20.
- Seq 4.21 **[Req 363]** Interpret and place the carriage return on the screen when provided in the CRes message by the ACS.
- Seq 4.22 **[Req 364]** Not display any UI elements other than those provided in the CRes message by the ACS in the Branding, Challenge/Processing and Information zones.
- Seq 4.23 **[Req 365]** Display the Expandable Information Label as a graphical control element that can be expanded (for example, an accordion).
- Seq 4.24 **[Req 366]** Display the Expandable Information Text only when the Cardholder selects the Expandable Information Label.
- Seq 4.25 **[Req 367]** Display the Why Information Label as a graphical control element that can be expanded (for example, an accordion).
- Seq 4.26 **[Req 368]** Display the Why Information Text only when the user selects the Why Information Label.
- Seq 4.27 **[Req 369]** Provide the Cancel action. This can be implemented as a button in the top corner of the header zone as depicted in Figure 4.1 and Figure 4.2 and/or as a function on a controller for the platform.
- Seq 4.28 **[Req 392]** Display the Trust List Information Text with the default setting = Off so that a Cardholder action is required to generate a Y value in Trust List Data Entry.
- Seq 4.29 **[Req 446]** Display the Device Binding Information Text with the default setting = Off so that a Cardholder action is required to generate a Y value in Device Binding Data Entry.

The ACS shall for the CReq/CRes message exchange:

- Seq 4.30 **[Req 387]** Include only the ACS-chosen UI data elements supported for the selected ACS UI Type as defined in Table A.20.
- Seq 4.31 **[Req 370]** If used, represent a carriage return as specified in Table A.1 for the following data elements:
- Challenge Information Text
 - Expandable Information Text
 - Why Information Text
- Seq 4.32 **[Req 445]** If used, represent a bold text as specified in Table A.1 for the following data elements:
- Challenge Information Text
 - Expandable Information Text
 - Why Information Text
- Seq 4.33 **[Req 429]** Include the image in either png or jpeg format when providing the Issuer Image and Payment System Image.

4.2.3 Native UI Templates

Figure 4.14–Figure 4.37 depict sample Native UI templates. The UI content is provided by the ACS in the CRes message which contains the information needed to properly display the UI.

UI elements exceptions are depicted in the figures as “Label Managed by the 3DS Requestor” and the functionality of these items are managed by the 3DS SDK.

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

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

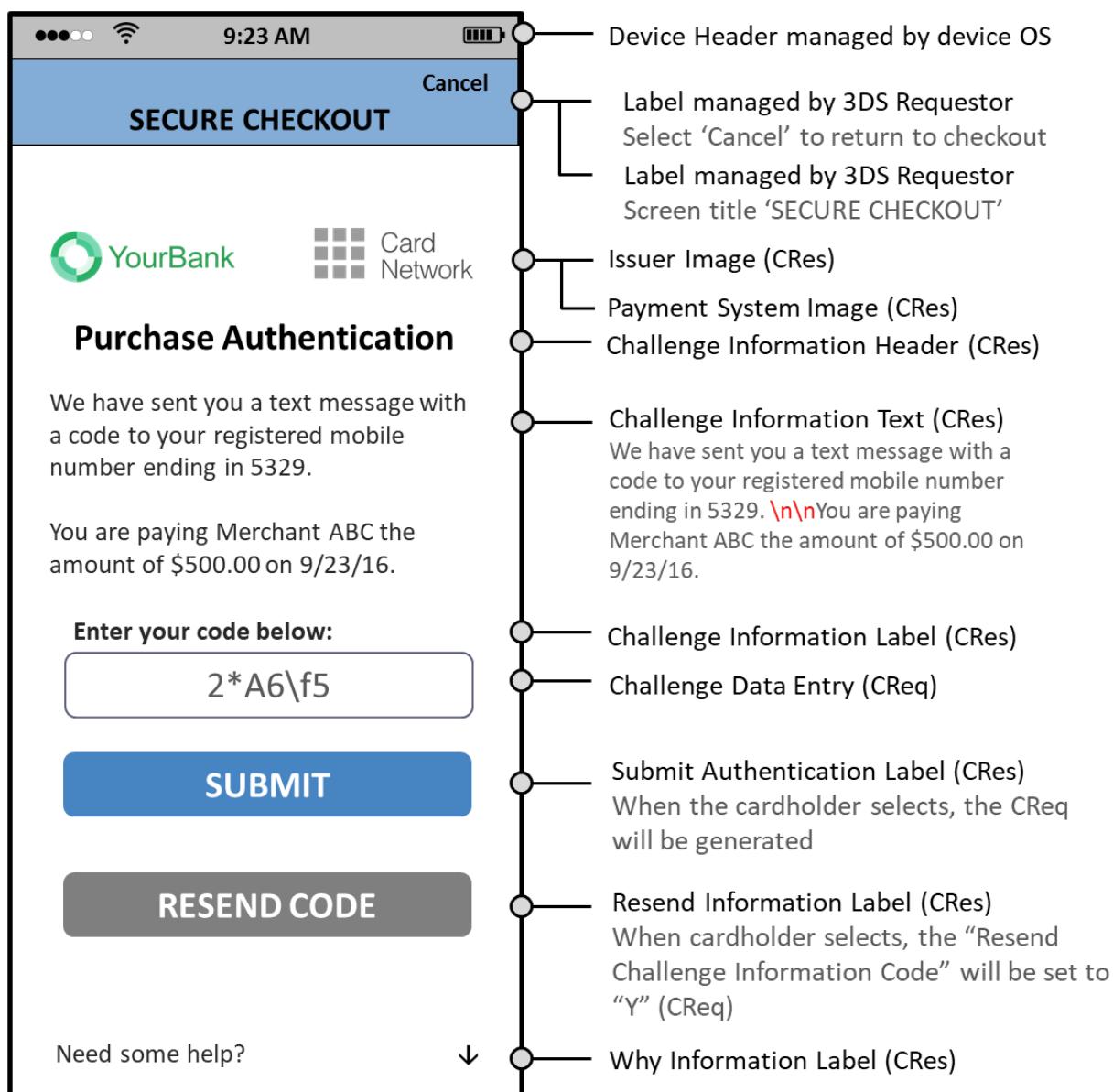


Figure 4.15: Sample Native UI OTP/Text Template—PA—Landscape

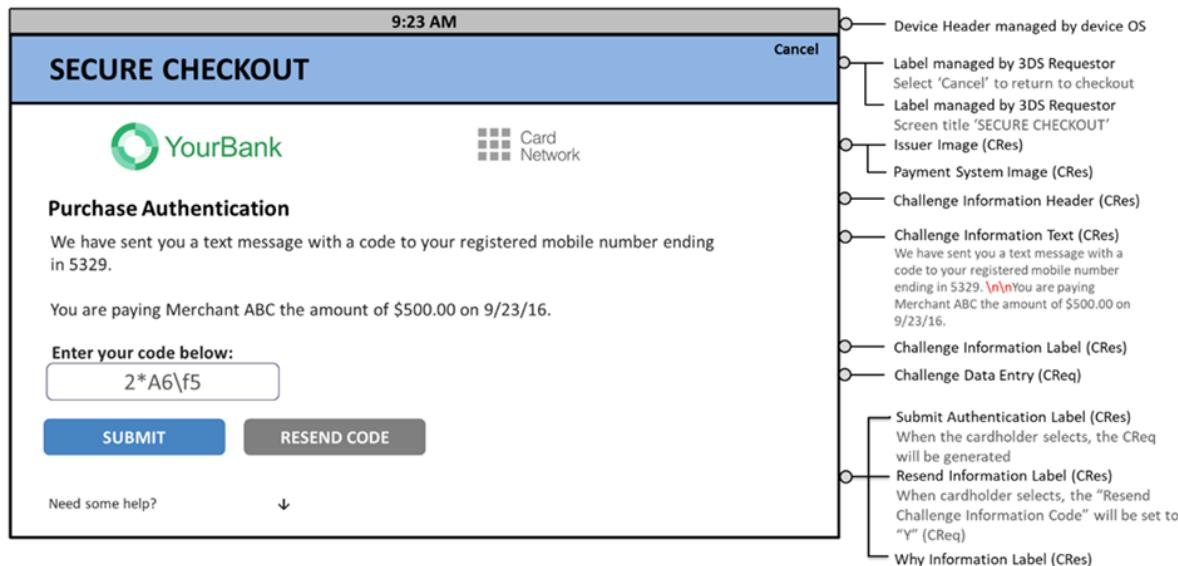


Figure 4.16 and Figure 4.17 provide sample formats with the optional second one-time passcode (OTP)/Text during a Payment Authentication transaction. This sample UI provides a format using expandable fields for additional information .

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

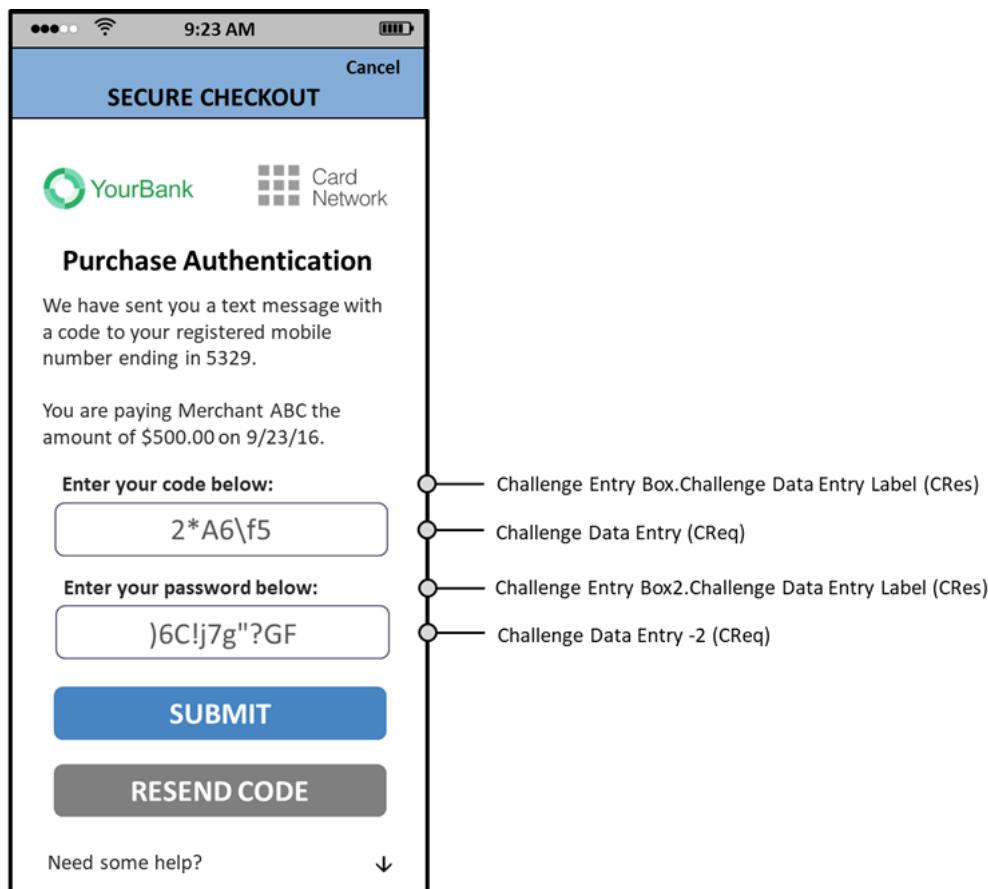


Figure 4.17: Sample Native UI with Optional Second OTP/Text entries Template—PA—Landscape

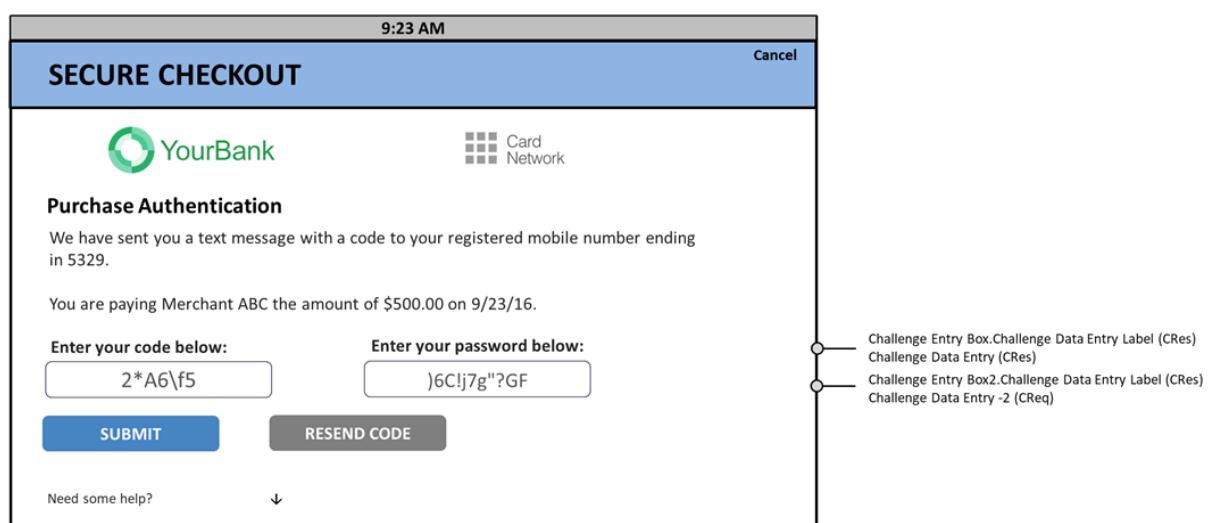


Figure 4.18 provides a sample format for a one-time passcode (OTP)/Text during a Non-Payment Authentication transaction.

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

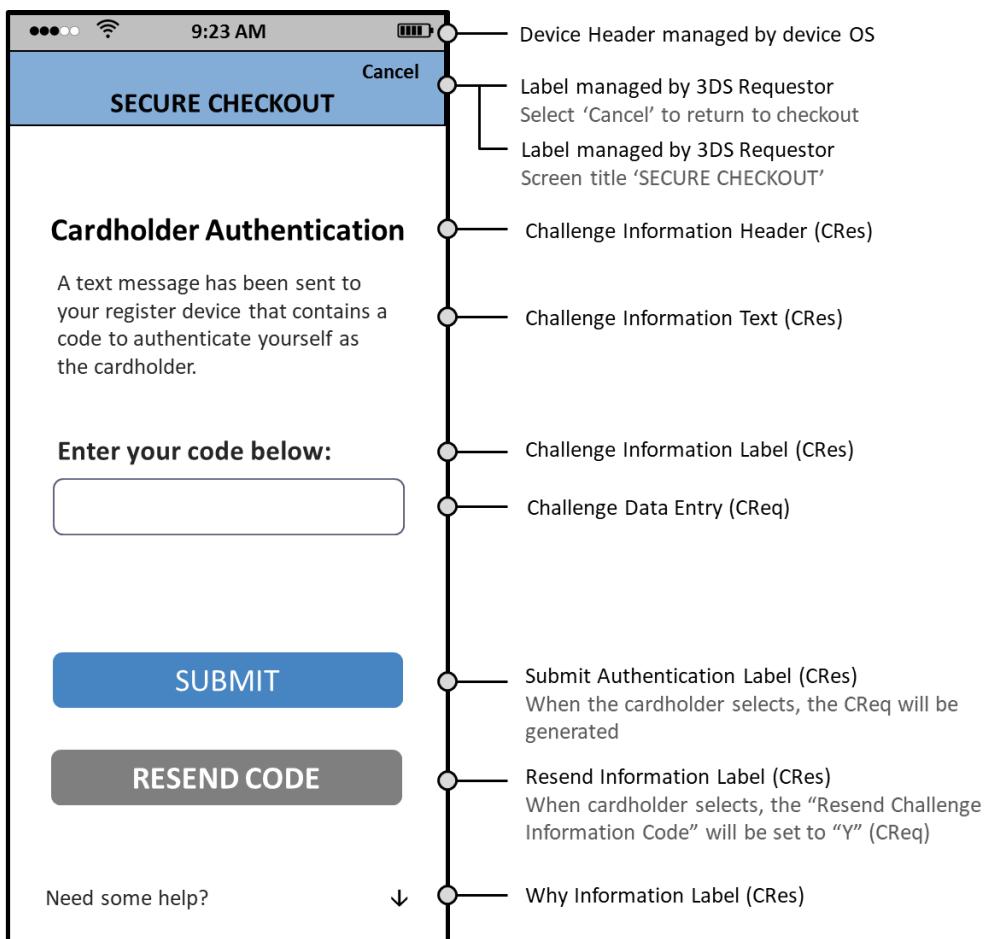


Figure 4.19 and Figure 4.20 provide sample formats that allows multiple options to be presented to the Cardholder to obtain single response. For example, asking the Cardholder if they prefer the OTP to be sent to the Consumer Device or to the email address on file.

Note: To optimise the Cardholder experience, the Challenge Selection Information can be displayed horizontally or vertically in landscape.

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

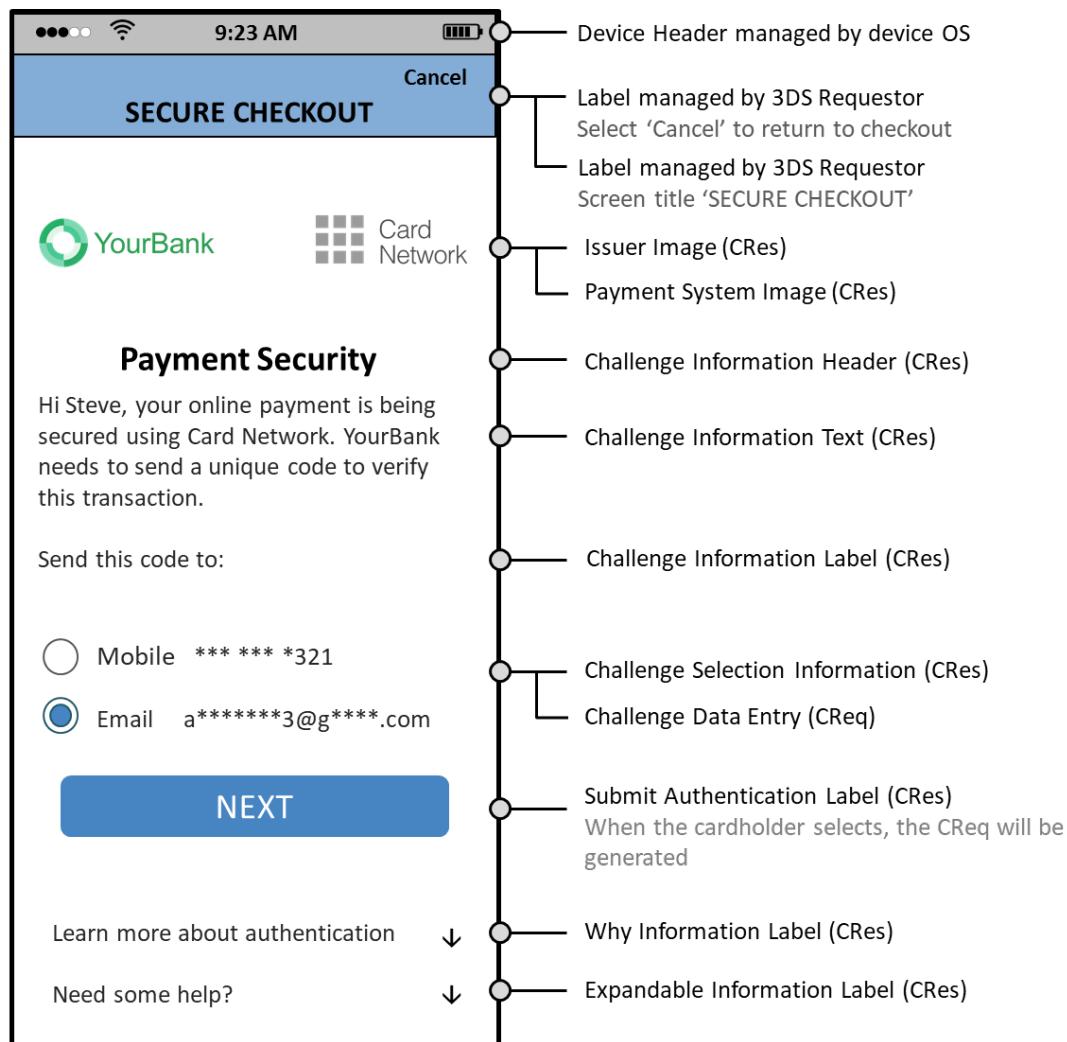


Figure 4.20: Sample Native UI—Single-select Information—PA—Landscape

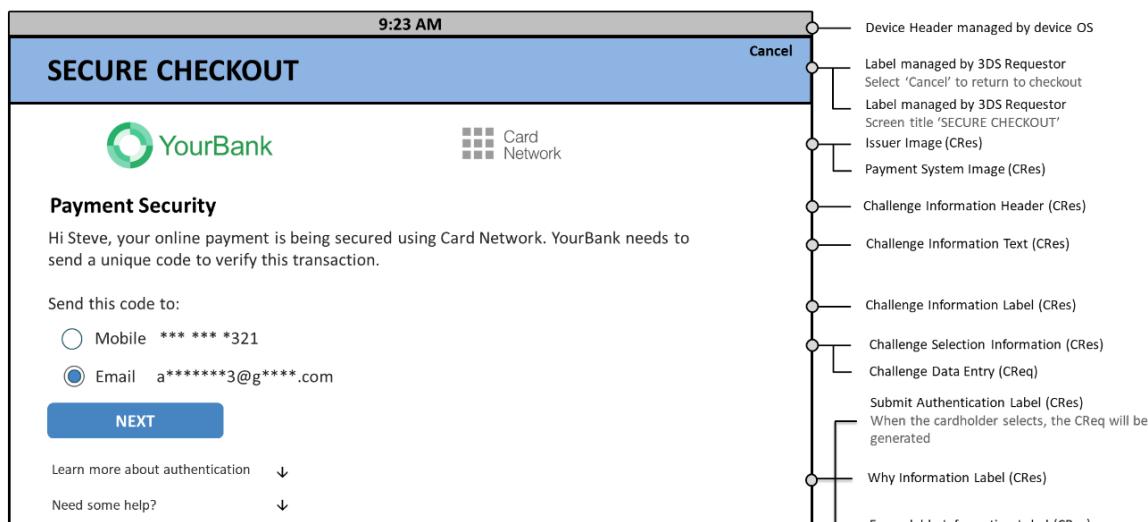


Figure 4.21 and Figure 4.22 provide sample formats that allows multiple options to be presented to the Cardholder to obtain multiple responses on a single screen. For example, asking the Cardholder to select the cities where they have lived. This example also depicts a screen with no Issuer or Payment System branding.

Note: To optimise the Cardholder experience, the Challenge Selection Information can be displayed horizontally or vertically in landscape.

Figure 4.21: Sample Native UI—Multi-select Information—PA—Portrait

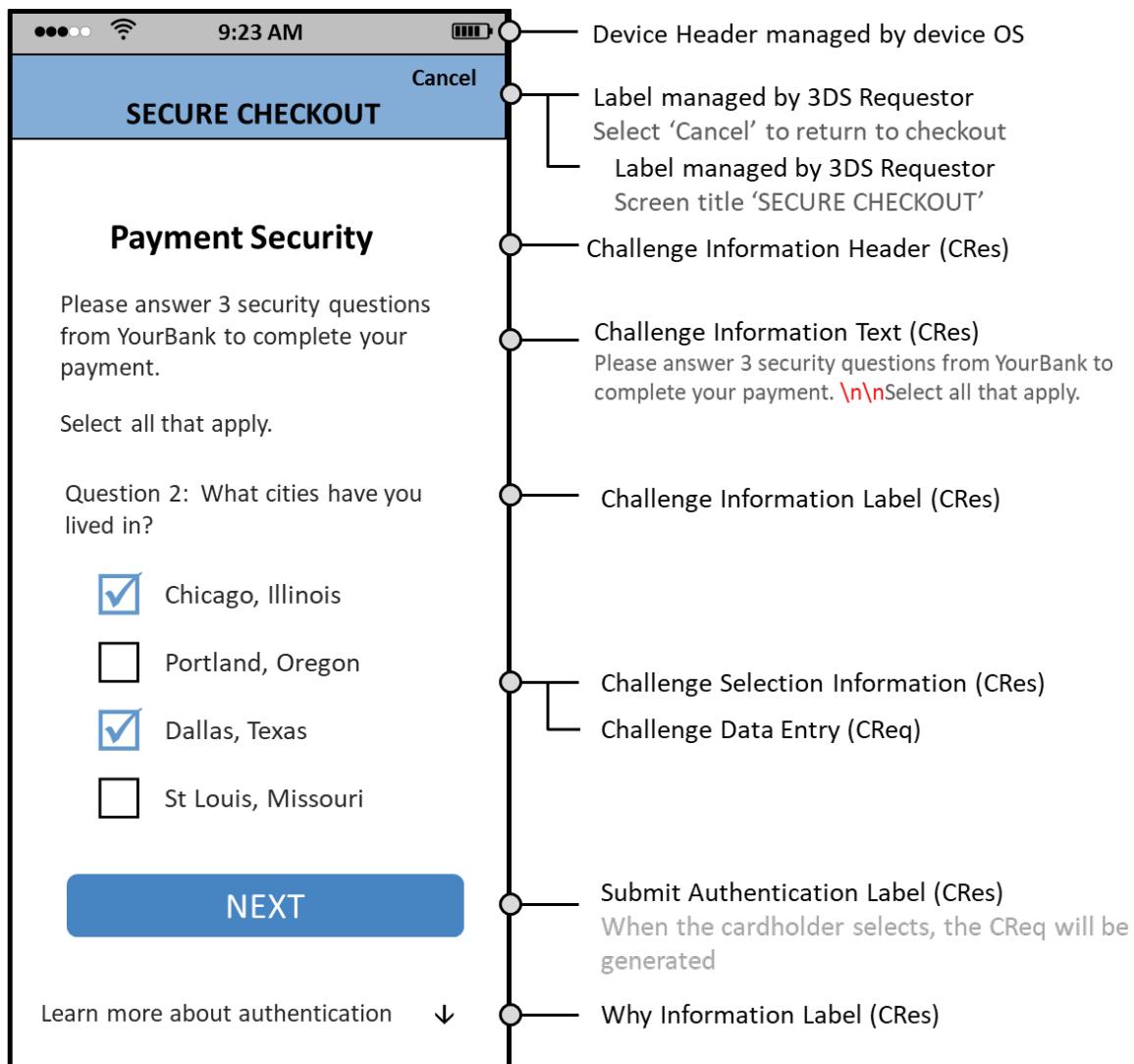
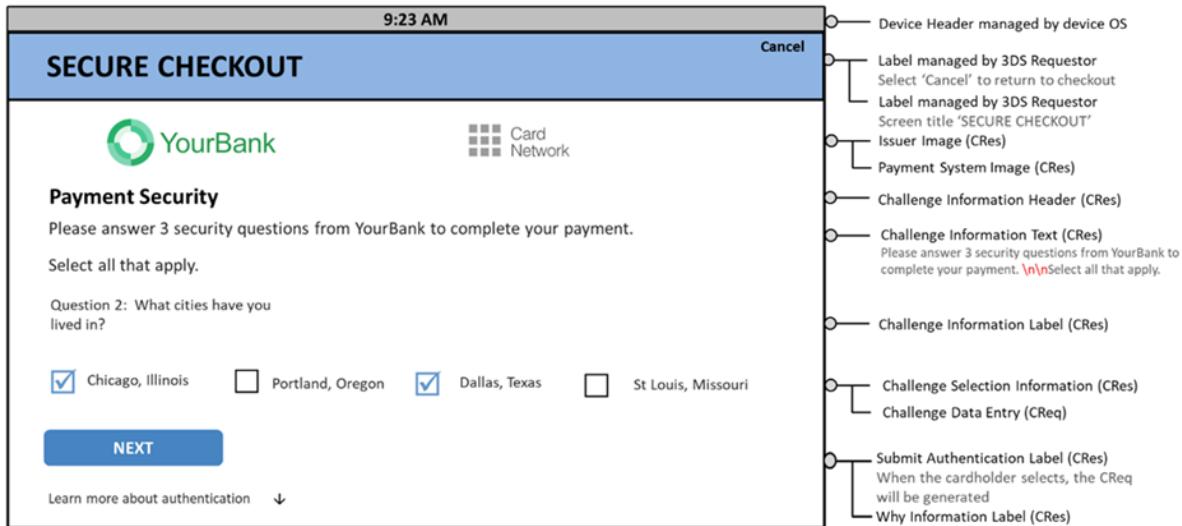


Figure 4.22: Sample Native UI—Multi-select Information—PA—Landscape



The Out-of-Band (OOB) user interface allows Issuers to utilise authentication methods other than dynamic and static data such as an Issuer's mobile app. When an OOB challenge is necessary, the Issuer/ACS provides instructions to the Cardholder to explain the authentication process.

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

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

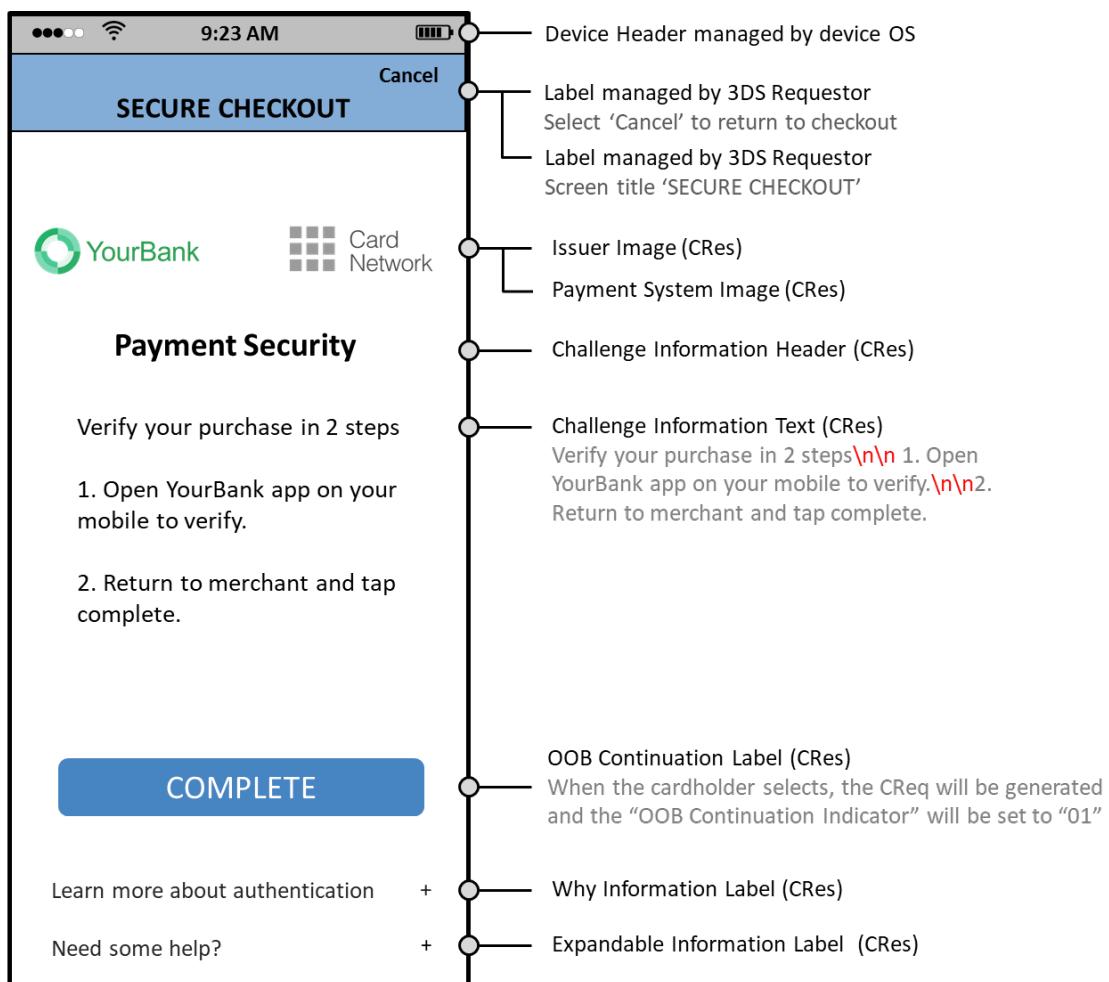


Figure 4.24: Sample OOB Native UI Template with Complete button—PA—Landscape

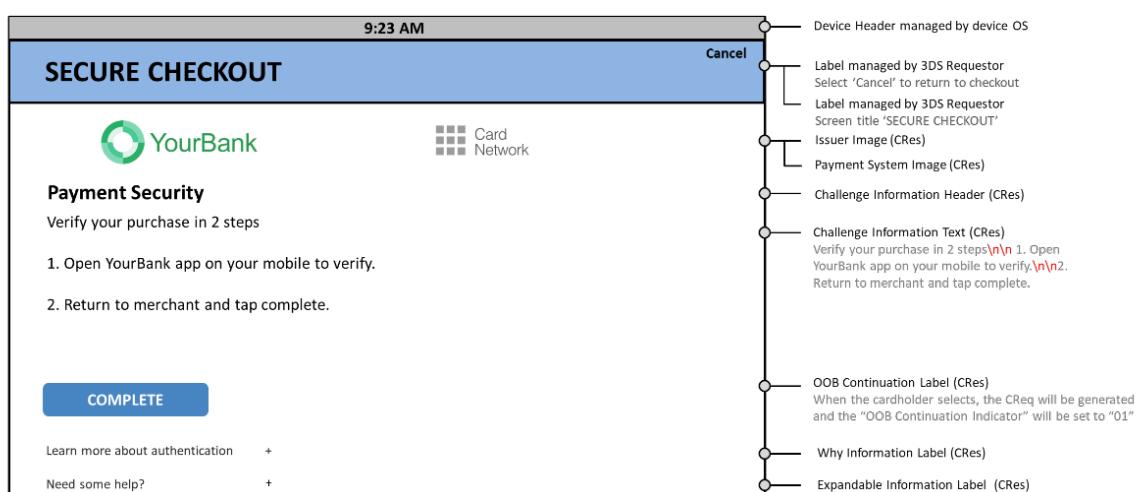


Figure 4.25 and Figure 4.26 provide sample OOB formats that display a button to open an authentication App.

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

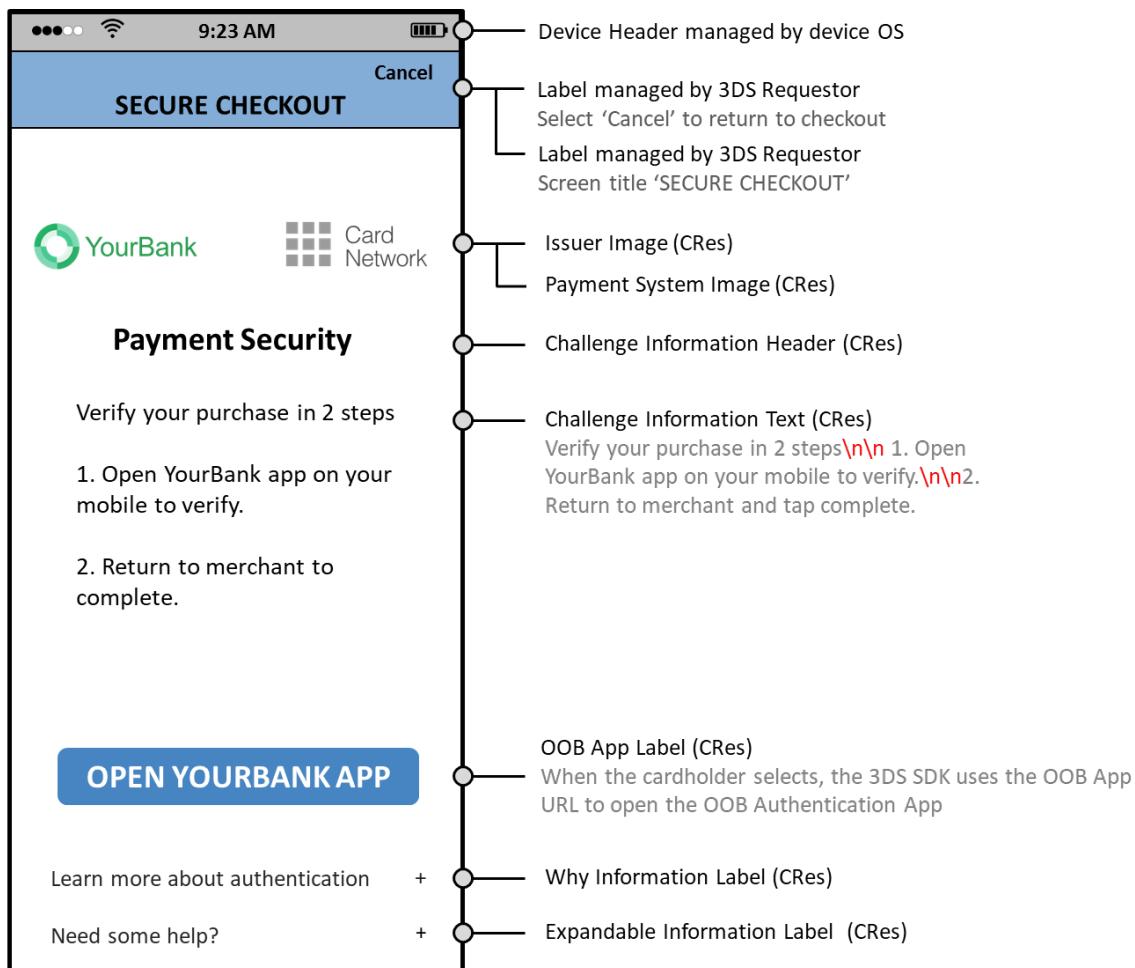


Figure 4.26: Sample OOB Native UI Template with Automatic OOB App URL link—Landscape

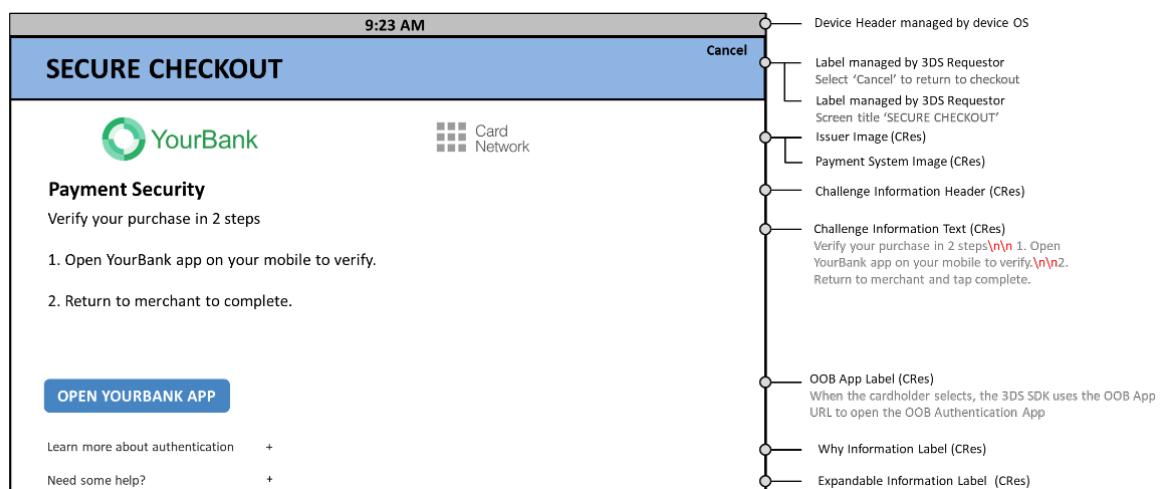


Figure 4.27 provides a sample of how the Challenge Information Text Indicator may be displayed to the Cardholder.

Figure 4.27: Sample Challenge Information Text Indicator—PA

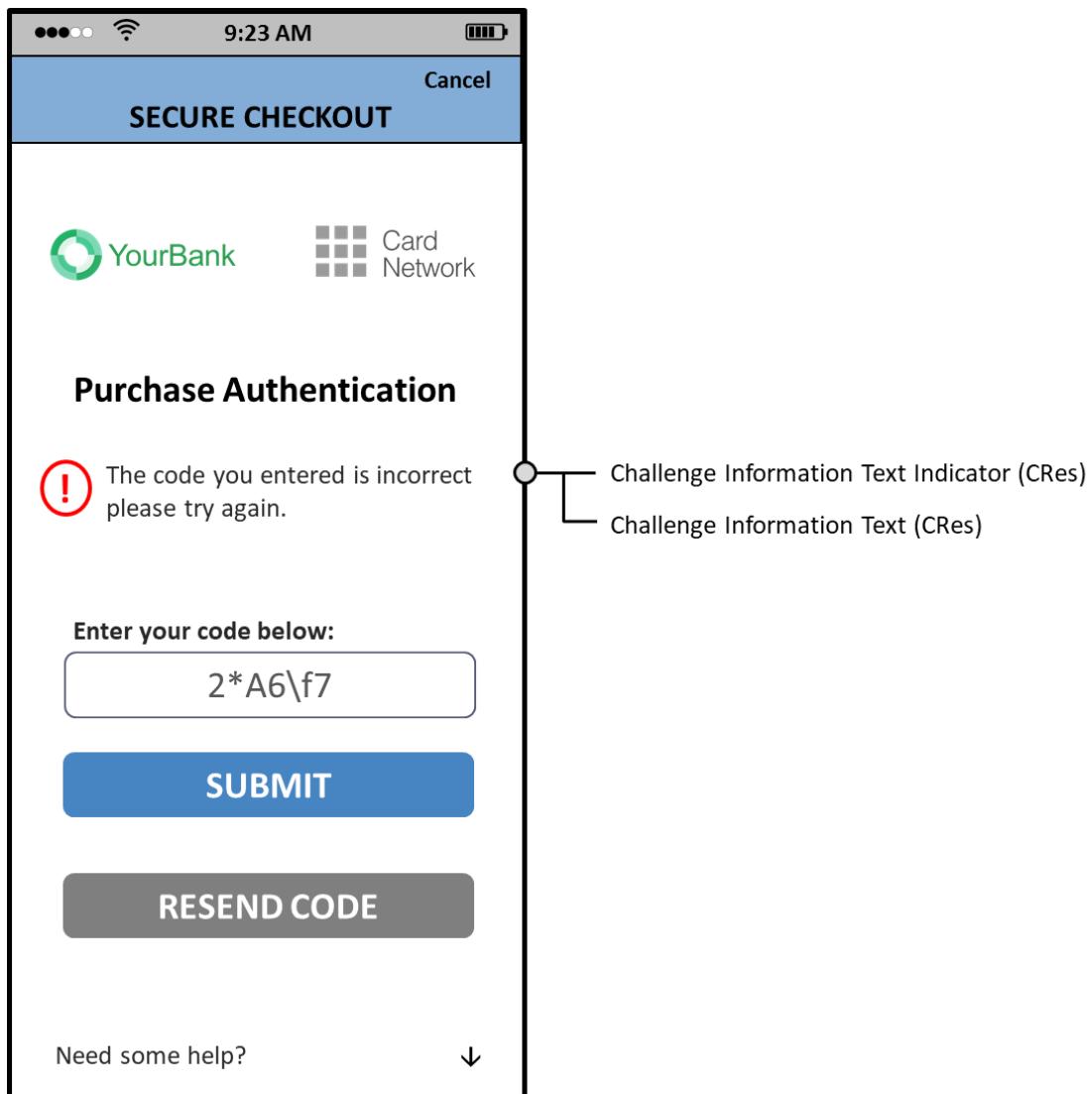


Figure 4.28–Figure 4.33 provide sample formats that display the two possible positions for the Trust List option and Device Binding option (above or below the buttons) as defined in Table A.20.

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

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

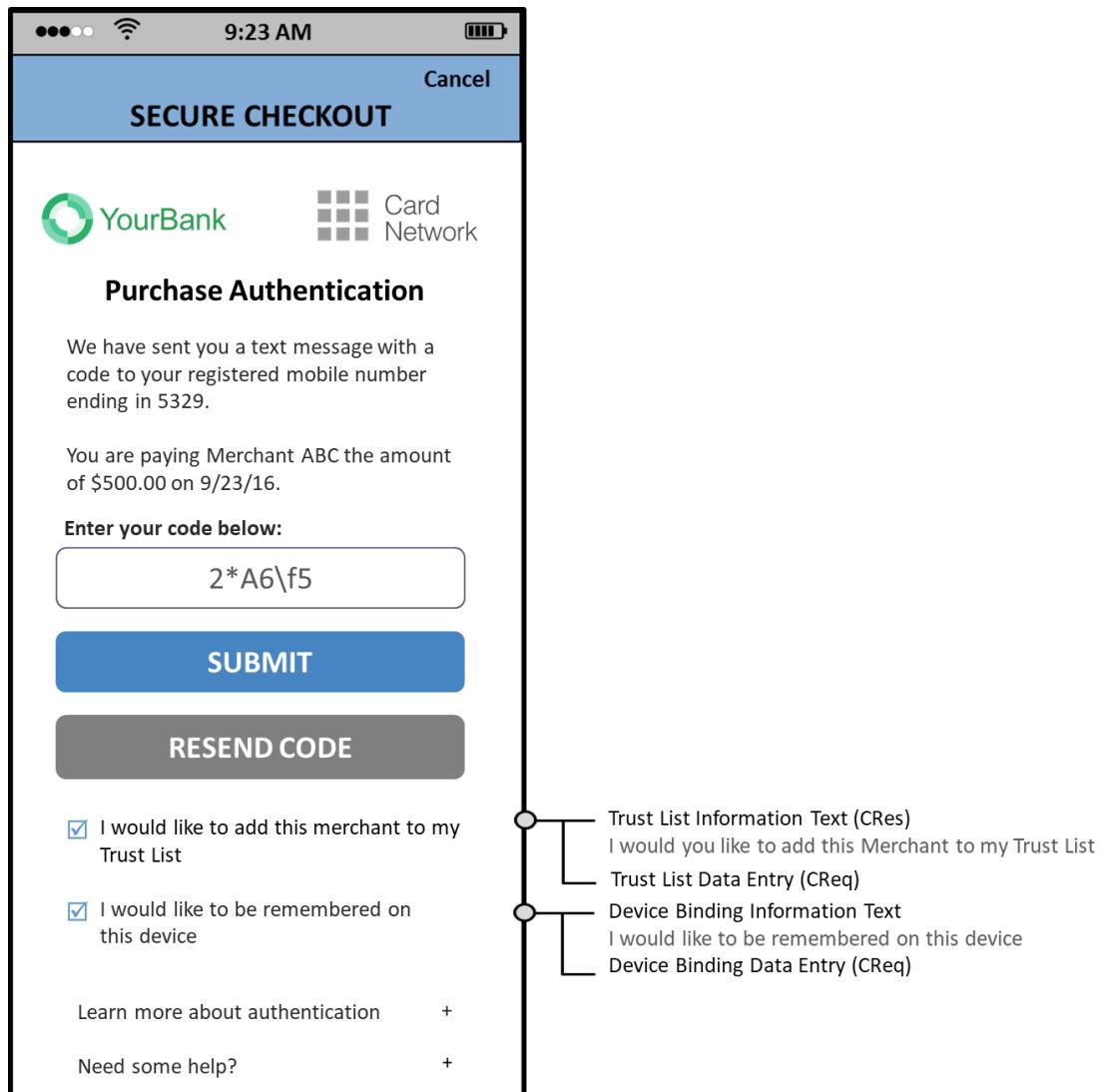


Figure 4.29: Sample Trust List/Device Binding Information Text—PA—Portrait

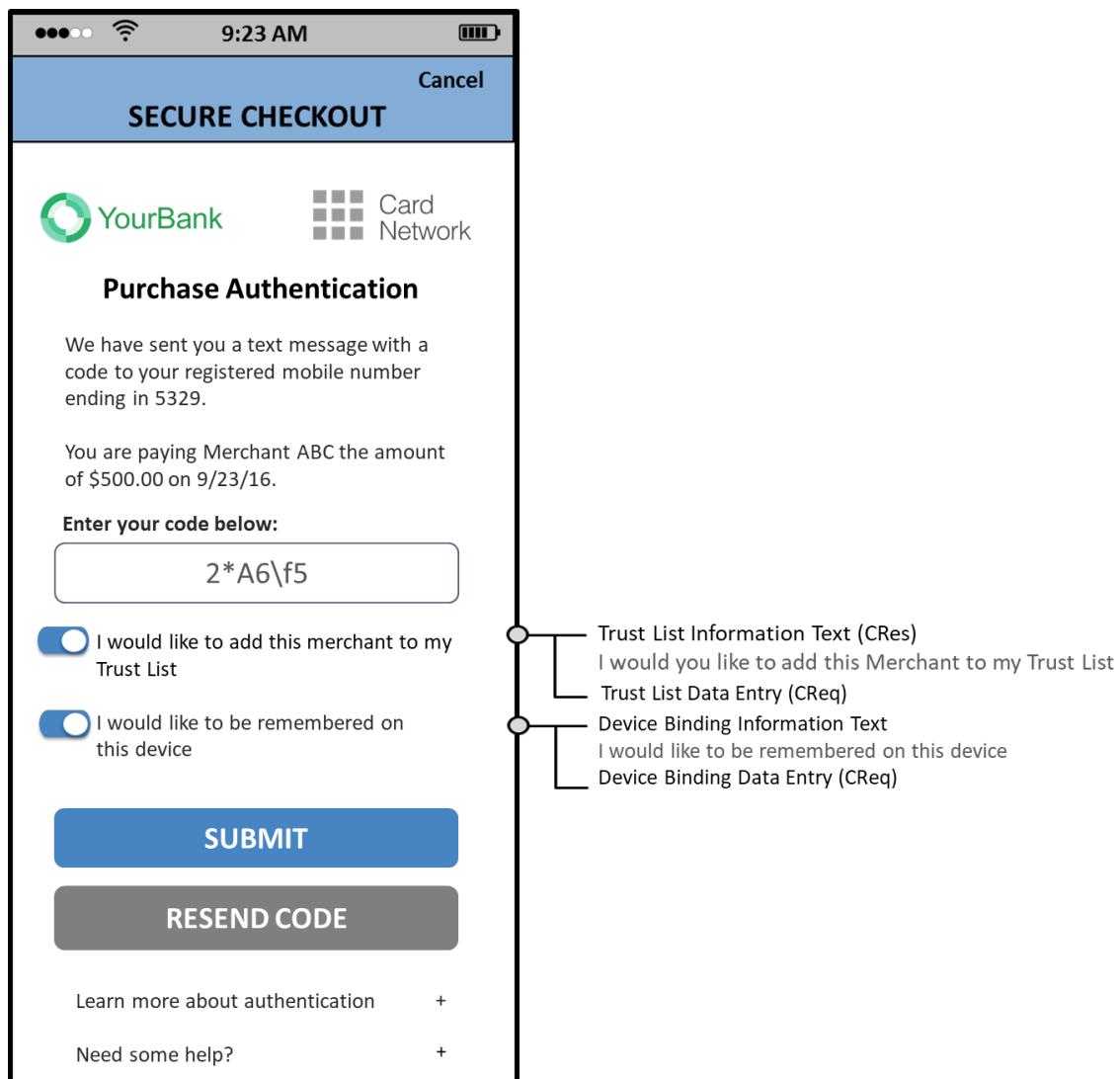


Figure 4.30: Sample Trust List/Device Binding Information Text—PA—Landscape

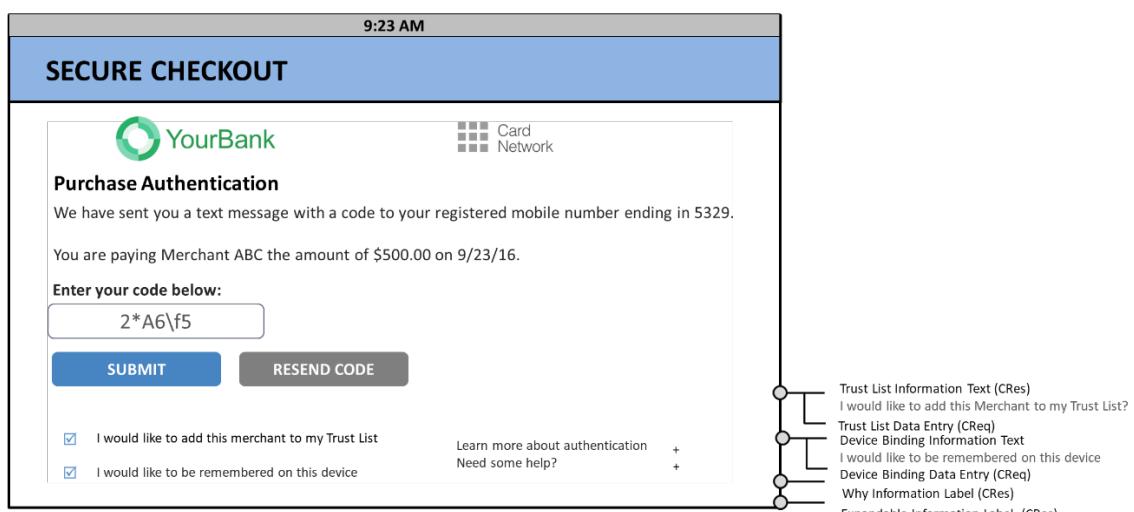


Figure 4.31: Sample Trust List/Device Binding Information Text—PA—Landscape

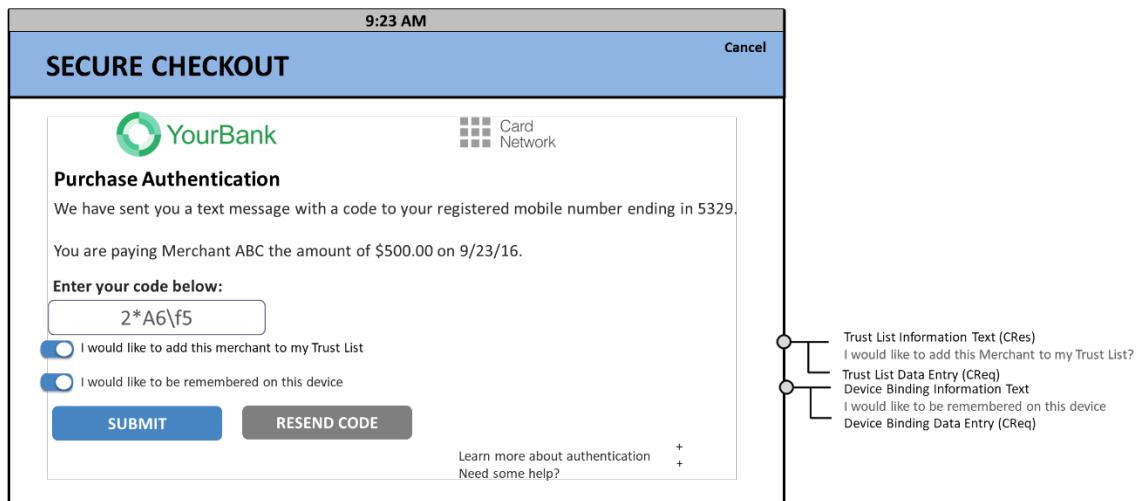


Figure 4.32 and Figure 4.33 provide sample UI formats to display instructions to the Cardholder.

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

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

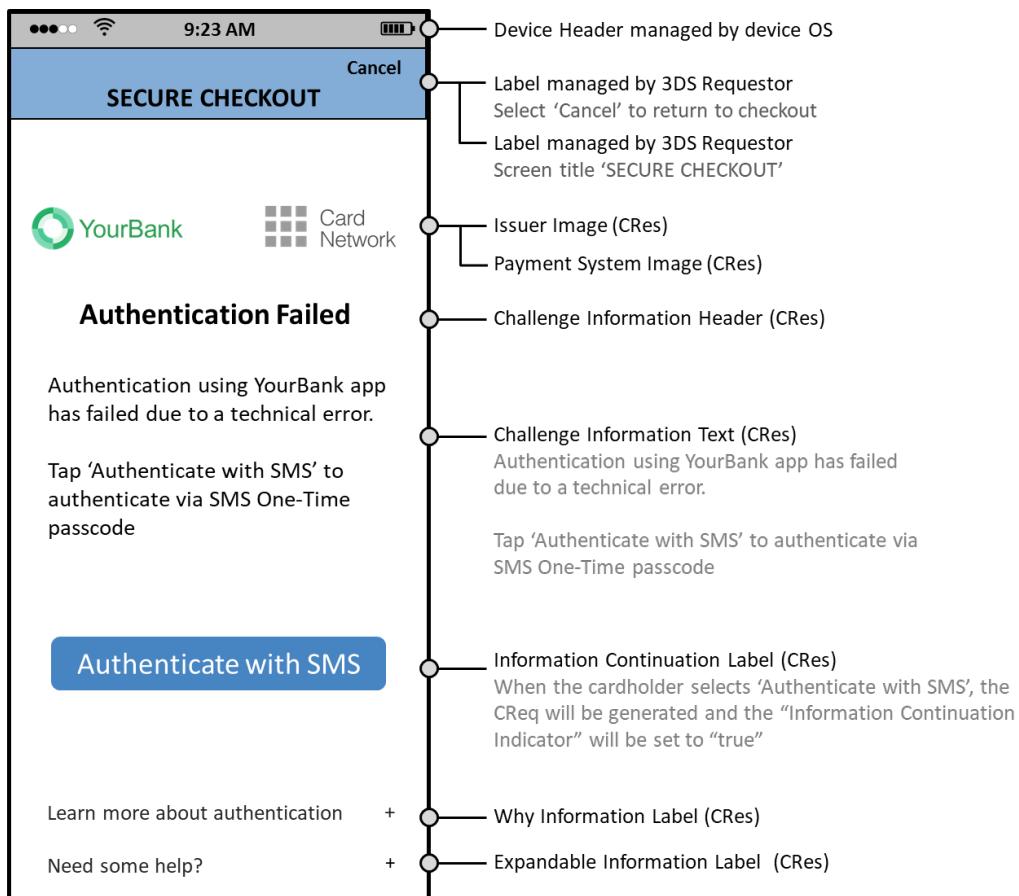


Figure 4.33: Sample Information Native UI Template—PA—Landscape

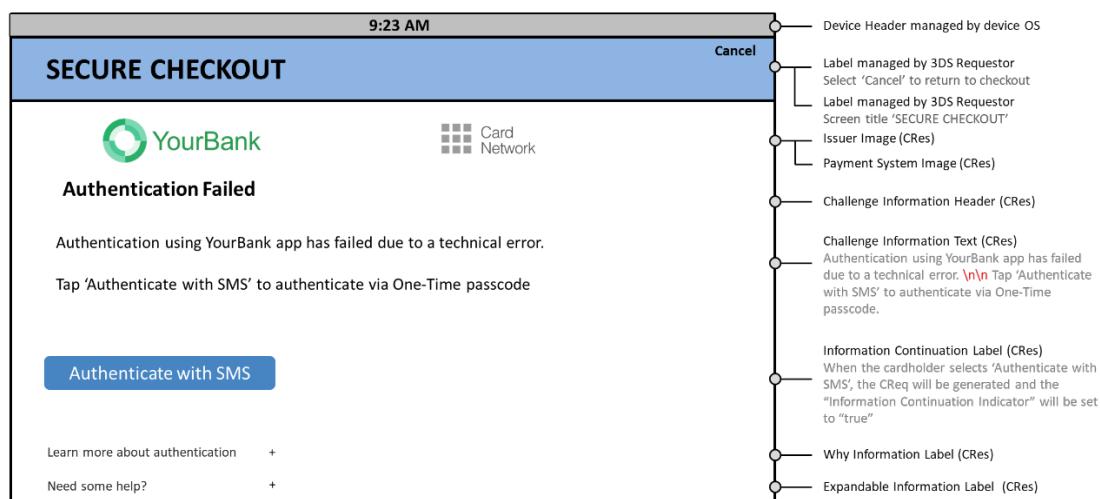


Figure 4.34 and Figure 4.35 provides a sample format that uses the Challenge Data Entry Masking and Challenge Data Entry Masking Toggle options during a purchase authentication.

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

Figure 4.34: Sample Challenge Data Entry Masking—PA

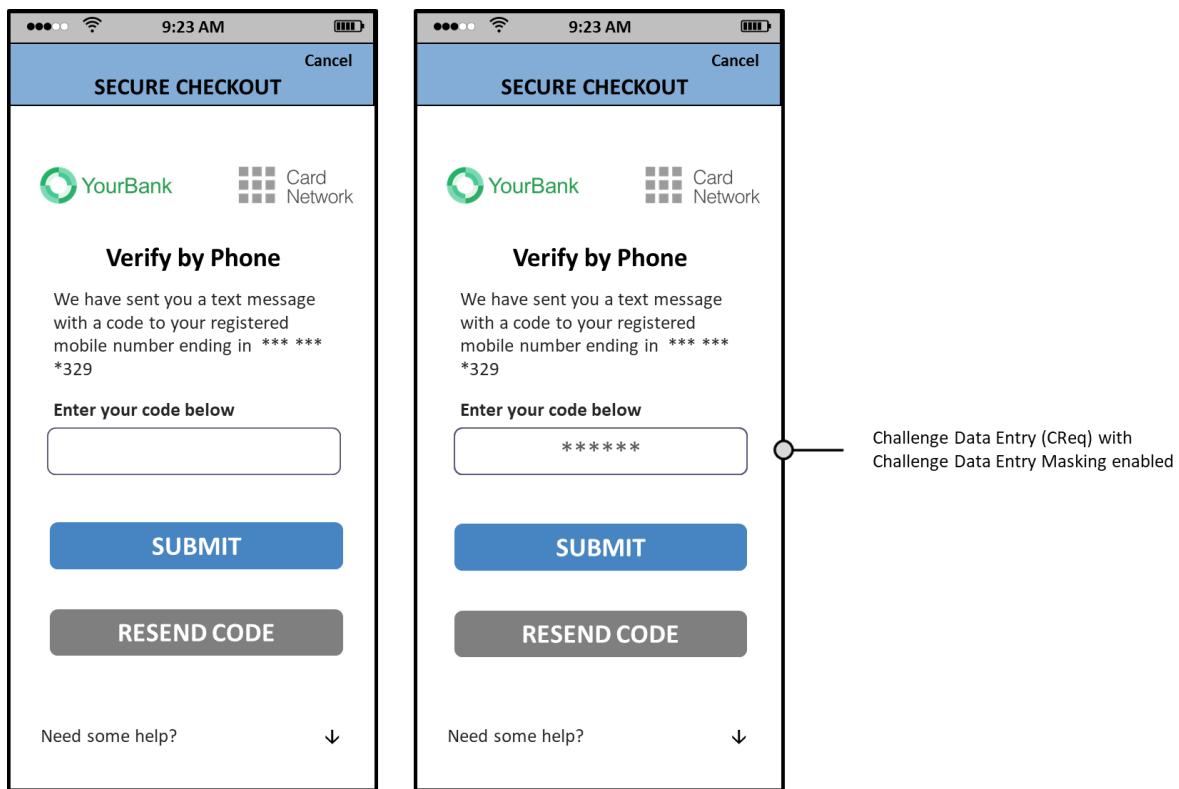


Figure 4.35: Sample Data Entry Masking with Toggle

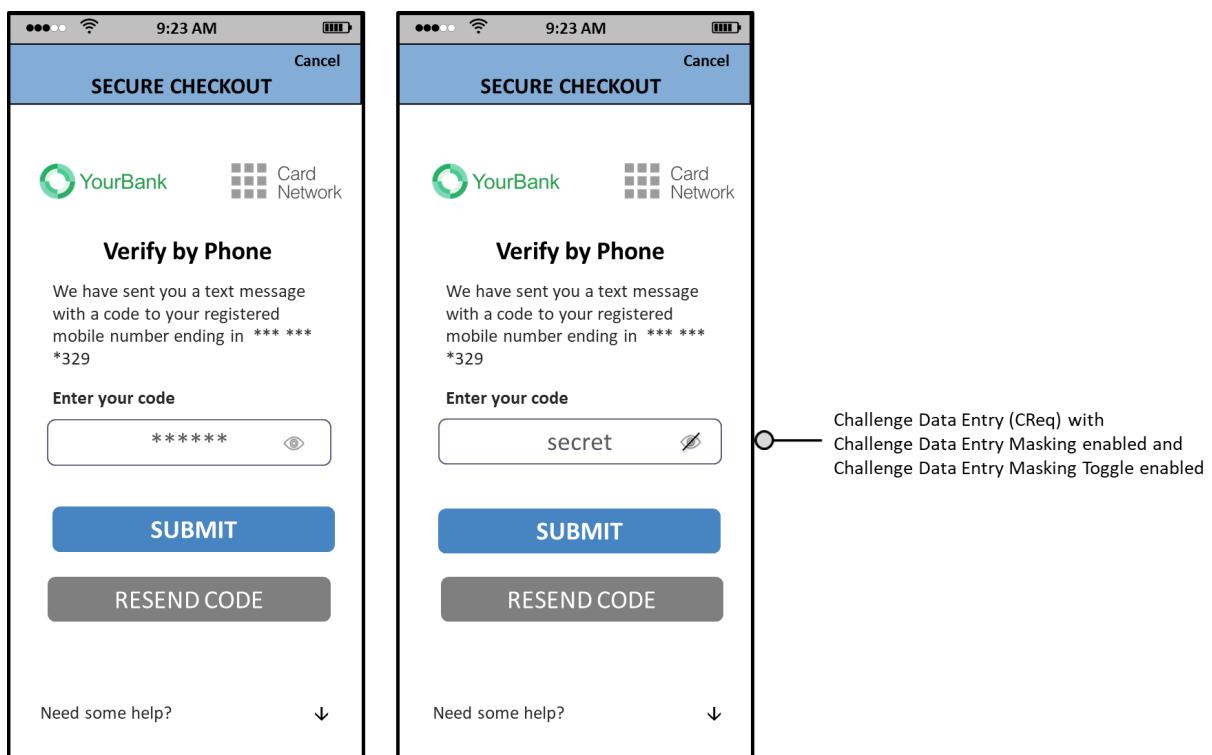


Figure 4.36 and Figure 4.37 provide sample formats with the Challenge Additional Label which provides additional flexibility in challenge management available for all UI templates.

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

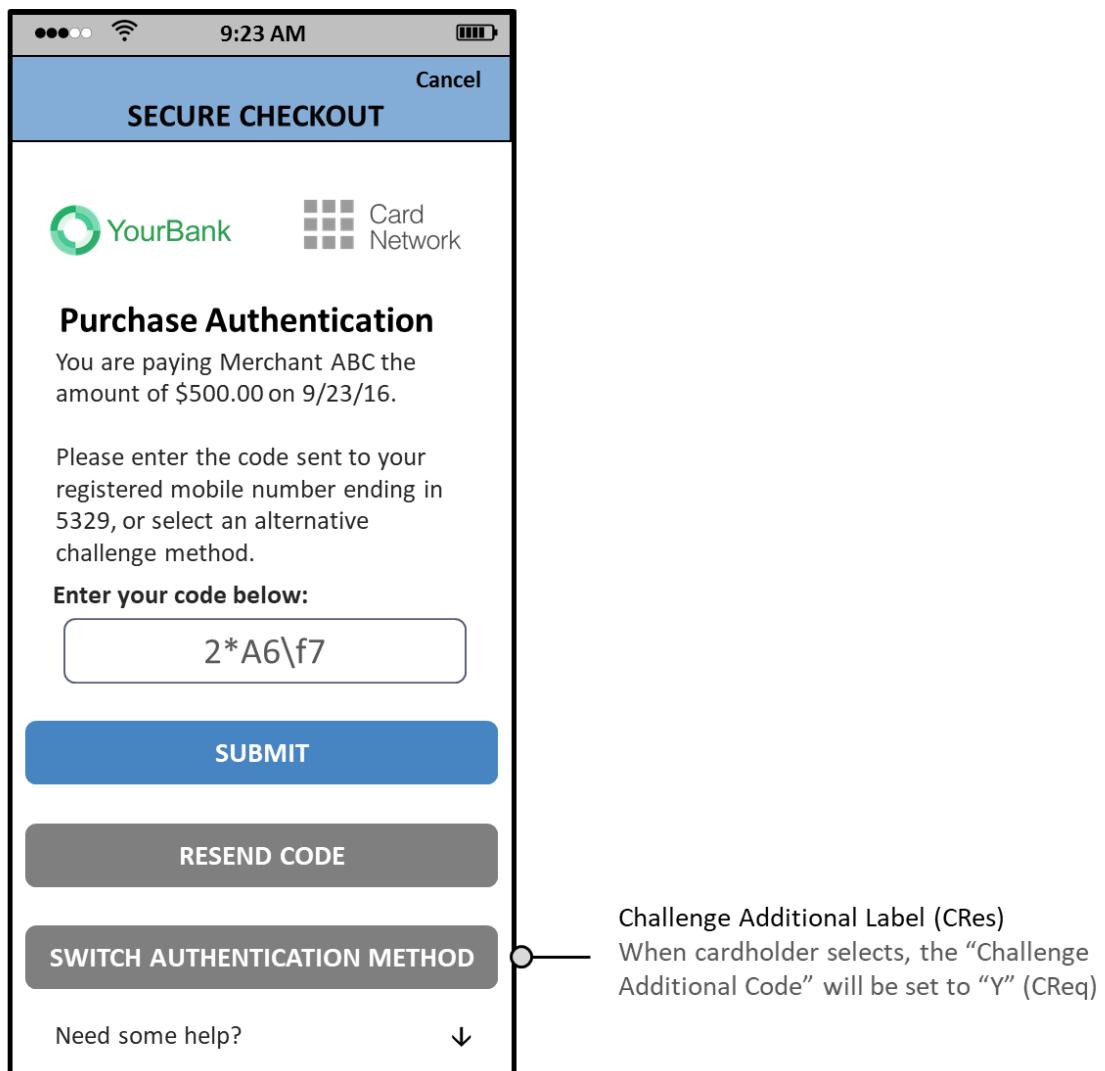


Figure 4.37: Sample Native UI OTP/Text Template with Challenge Additional Label—PA—Landscape

The screenshot shows a mobile application interface titled "SECURE CHECKOUT". At the top, it displays the time "12:29 PM". Below the title, there are two logos: "YourBank" and "Card Network". The main content area is titled "Purchase Authentication" and contains the following text:
"You are paying Merchant ABC the amount of \$500.00 on 9/23/20.
Please enter the code sent to your registered mobile number ending in 5329,
or select an alternative challenge method."
Below this, there is a text input field labeled "Enter your code below:" containing the code "2*A6\f5". At the bottom of the screen are three buttons: "SUBMIT", "RESEND CODE", and "SWITCH AUTHENTICATION METHOD". A link "Need some help?" is located next to the "RESEND CODE" button. An arrow points from the "SWITCH AUTHENTICATION METHOD" button to a callout box on the right. The callout box contains the text: "Challenge Additional Label (CRes)
When cardholder selects, the "Challenge Additional Code" will be set to "Y" (CReq)".

4.2.4 Native UI Message Exchange Requirements

The CReq/CRes message exchange is the same for both Native and HTML, however, there is a difference in the data elements and templates. This section identifies the requirements for an App-based Native UI.

4.2.4.1 3DS SDK

The 3DS SDK shall:

- Seq 4.34 **[Req 153]** After submitting the CReq message to the ACS, display the same Processing screen as during the AReq/ARes message until the CRes message is received, or timeout is exceeded. Refer to Section 5.5.2.2 for CReq/CRes message Timeout requirements.

4.2.4.2 ACS

As defined via the requirements in Section 3.1, the ACS will:

Use the secure CReq/CRes channel through the 3DS SDK for communication with the Consumer Device and populate the applicable ACS UI Type that the 3DS SDK will need to display as identified in **[Req 55]**.

The ACS will continue the Challenge message cycle until complete. Whether the cycle is completed is communicated to the 3DS SDK via the Challenge Completion Indicator data element in the CRes message.

The CRes message sent to the 3DS SDK will populate the appropriate data elements to render the screen as designed by the Issuer/ACS.

4.2.4.3 3DS SDK

The 3DS SDK shall:

- Seq 4.35 **[Req 154]** Act upon any action to exit the 3DS SDK (for example, the Cancel action on screen or through an external controller) and return control to the 3DS Requestor App.

After receiving the CRes message from the ACS, the 3DS SDK displays the requested content through the 3DS Requestor App.

The 3DS SDK shall:

- Seq 4.36 **[Req 155]** Control UI interaction processing, for example, the Cancel action.
- Seq 4.37 **[Req 156]** Have the option to display a screen title.
- Seq 4.38 **[Req 157]** Return control to the 3DS Requestor App when the Cancel is activated.
- Seq 4.39 **[Req 158]** Create and generate the CReq message to return to the ACS in response to Cardholder interaction with the UI.

Note: The CReq/CRes message exchange continues until the Challenge Completion Indicator in the CRes is set to Y by the ACS or until the 3DS SDK times out.

4.2.5 HTML UI Display Requirements

The HTML UI provides Cardholders with an Issuer-consistent App-based experience across Consumer Devices that are able to render HTML. The HTML UI templates provides Issuers the ability to include Issuer-specific design elements (for example, branding, colours, and/or fonts).

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

The HTML UI implementation establishes a client–server relationship between the ACS-provided HTML document loaded in a 3DS Requestor's web view and the 3DS SDK process itself. This is accomplished by intercepting remote URL requests issued by the web view, and handling them within the 3DS SDK, rather than allowing them to pass through to the Consumer Device operating system and hence on to the Internet. This has two effects:

- Prevents maliciously formed HTML within the web view flow from requesting external resources or redirecting to an external malicious site (for example, a phishing page).
- Changes the web view form into an extension of the 3DS SDK's UI, one that is defined by the remote ACS using HTML, rather than by the 3DS SDK or 3DS Requestor's App.

Key HTML UI considerations:

- The contents of the web view are received as responsive HTML directly from the ACS and displayed in the web view by the 3DS SDK. Navigation attempts from within the web view are captured by the 3DS SDK and processed internally, rather than being passed to the operating system and network stack. In addition to navigation attempts, the 3DS SDK also captures external resource requests (image loads, external .JS scripts, CSS, etc.).
- The web view element is not being utilised as a Browser, but as a UI element whose content is the HTML and Cascading Style Sheets (CSS) provided by the ACS.
- A secure communication channel is established between the Consumer Device and the ACS for routing all the network communications. By defining all interaction with the web view in terms of the 3DS SDK, it clearly indicates that the 3DS SDK defines and owns the UI, and that the 3DS Requestor's App is isolated from the challenge interaction.

Details of the HTML UI and the rendering process are separately described in the applicable 3DS SDK specification and in the documentation provided by each DS.

Note: App-based HTML will function on all Consumer Devices that support HTML display.

4.2.5.1 3DS SDK/ACS

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

- Seq 4.40 **[Req 371]** Display the HTML as provided by the ACS.
- Seq 4.41 **[Req 372]** Display only the UI elements provided by the ACS in the Branding, Challenge/Processing and Information zones.
- Seq 4.42 **[Req 373]** Provide the Cancel action. This can be implemented as a button in the top corner of the header zone as depicted in Figure 4.1 and Figure 4.2 and/or as a function on a controller for the platform.

Note: The functionality of UI elements in the header zone is managed by the 3DS SDK.

The ACS shall for the CReq/CRes exchange:

- Seq 4.43 **[Req 374]** Create HTML with form elements in the applicable zones as outlined in Figure 4.1 and Figure 4.2 to support both portrait and landscape UI templates. The expected format is outlined in section 4.2.6.

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

- Seq 4.44 **[Req 375]** Expandable Information Label for display as a graphical control element that can be expanded (for example, an accordion) in the Information zone.
- Seq 4.45 **[Req 376]** Expandable Information Text for display in the Information zone only when the Cardholder selects the Expandable Information Label.
- Seq 4.46 **[Req 377]** Why Information Label for display as a graphical control element that can be expanded (for example, an accordion) in the Information zone.
- Seq 4.47 **[Req 378]** Why Information Text for display in the Information zone only when the Cardholder selects the Why Information Label.

4.2.6 HTML UI Templates

The HTML UI templates provide the ACS the ability to include Issuer-specific design elements (e.g., branding, colours, fonts) as shown in the figures below. Figure 4.38 and Figure 4.39 provide sample Payment Authentication HTML OTP UI templates that include Issuer branding.

Figure 4.38: Sample HTML UI OTP/Text Template—PA—Portrait

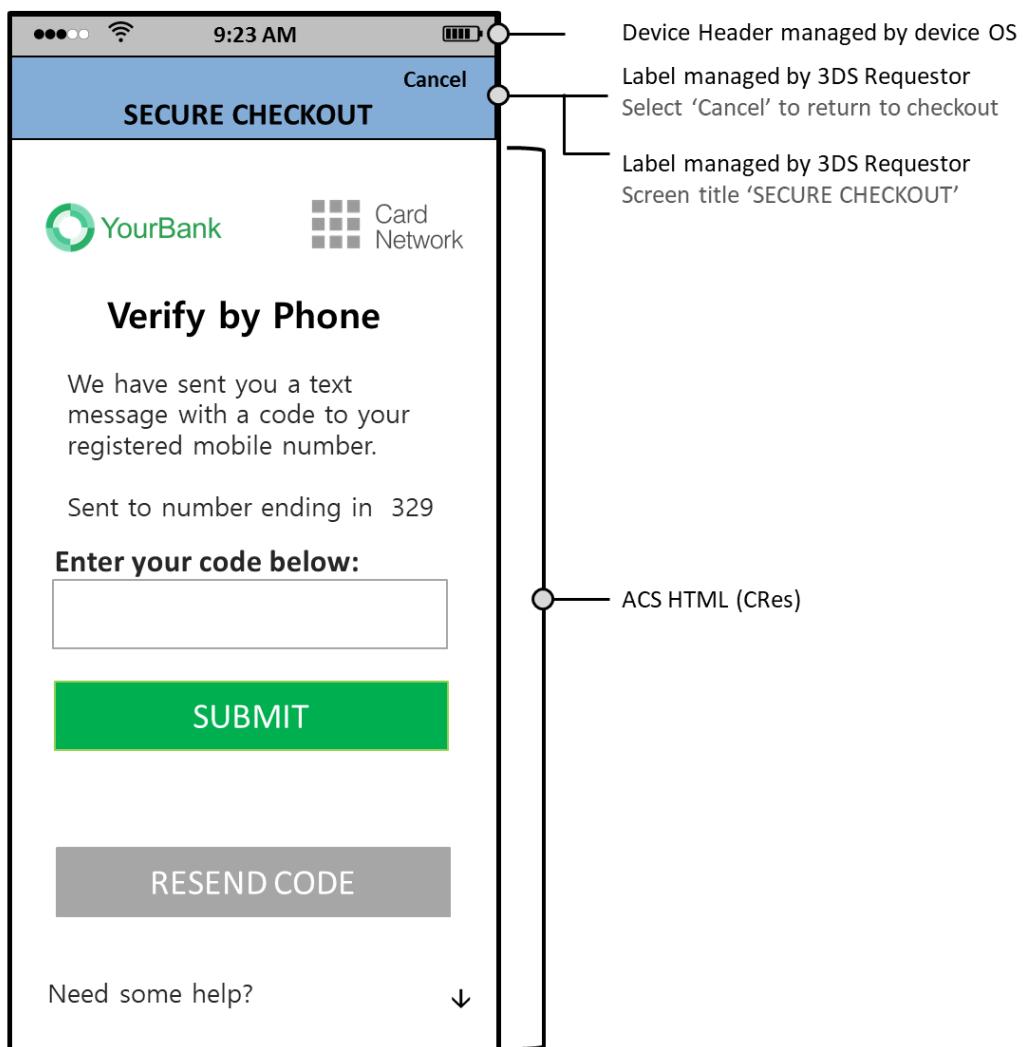


Figure 4.39: Sample HTML UI OTP/Text Template—PA—Landscape

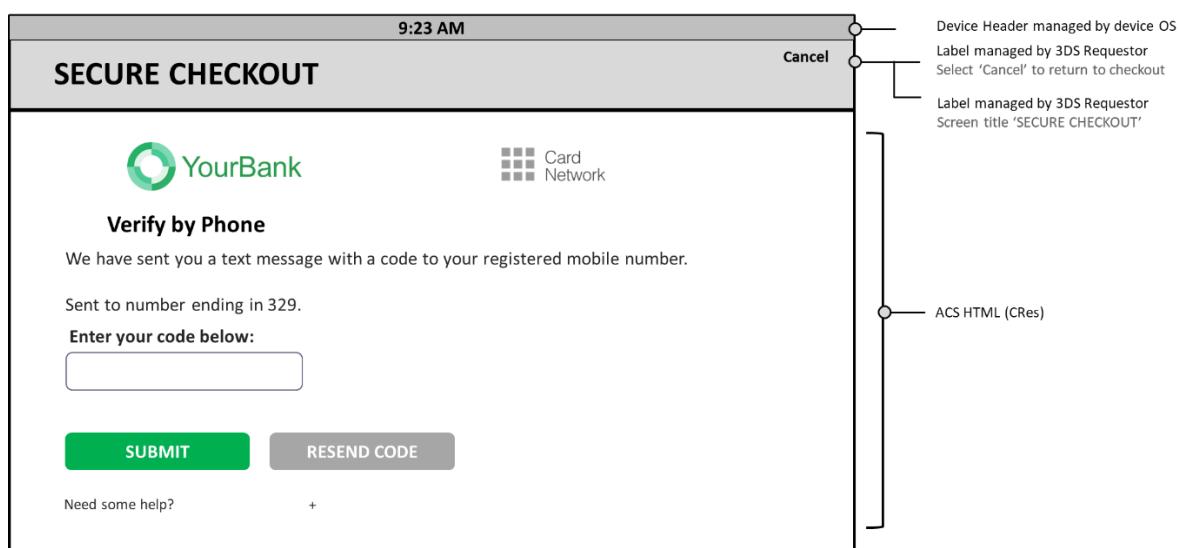


Figure 4.40 provides a sample Non-Payment Authentication HTML OTP UI template that includes Issuer branding. The HTML templates for Single-select and Multi-select are visually similar to the Native UI and therefore not repeated.

Figure 4.40: Sample UI OTP/Text Template—NPA

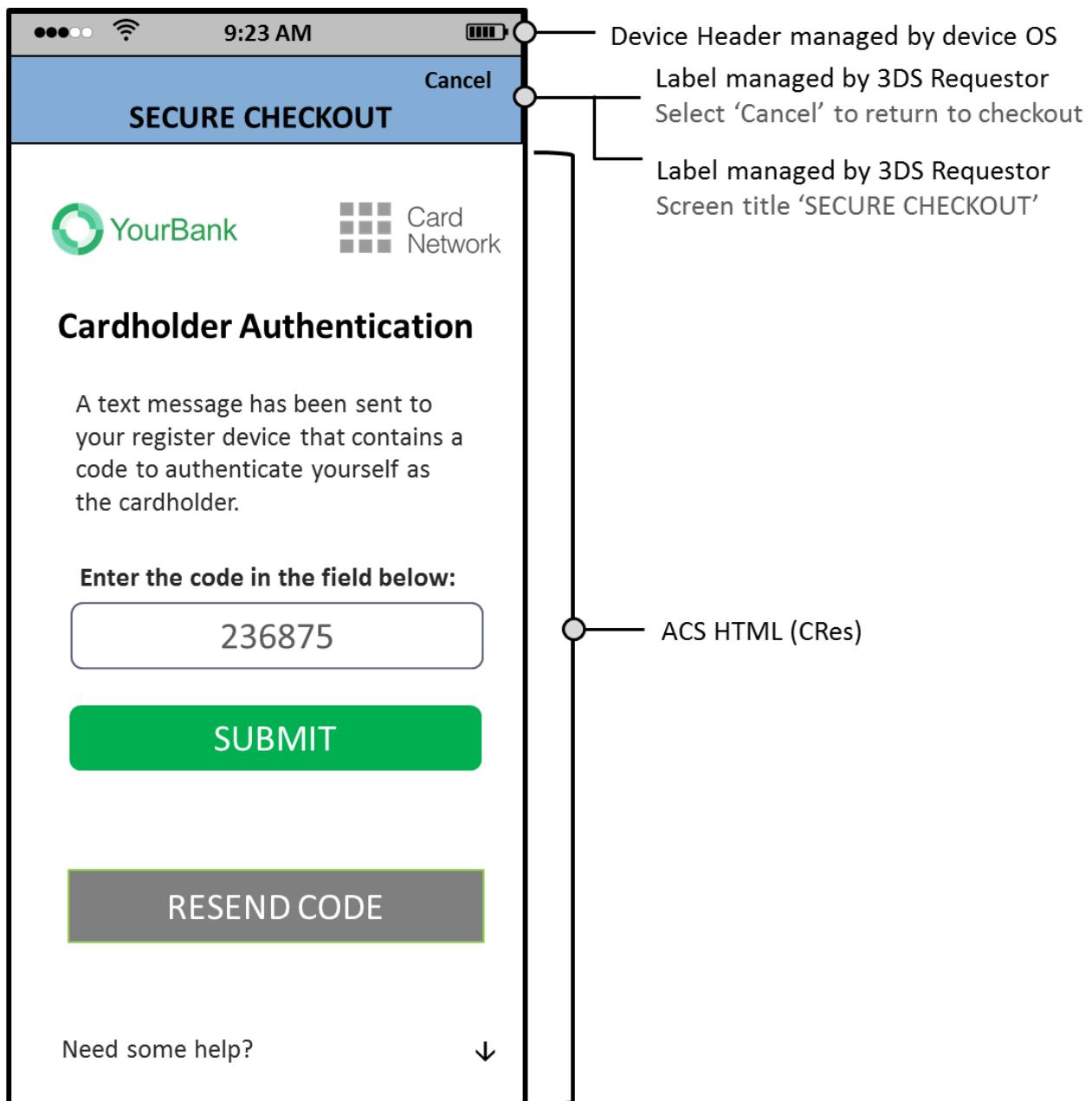


Figure 4.41–Figure 4.44 provide sample templates illustrating the OOB HTML UI.

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

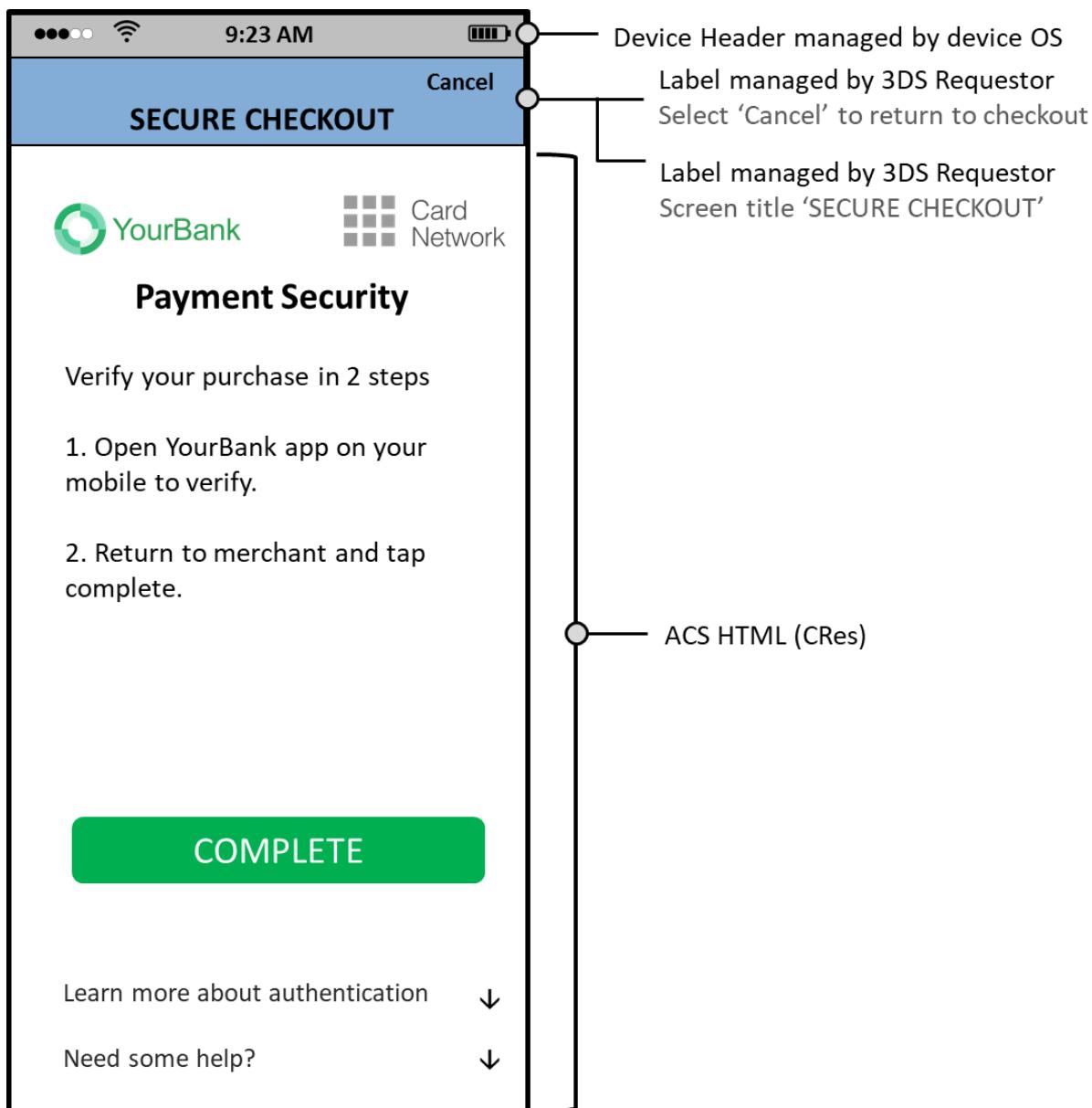


Figure 4.42: Sample OOB HTML UI Template with Complete button—PA—Landscape

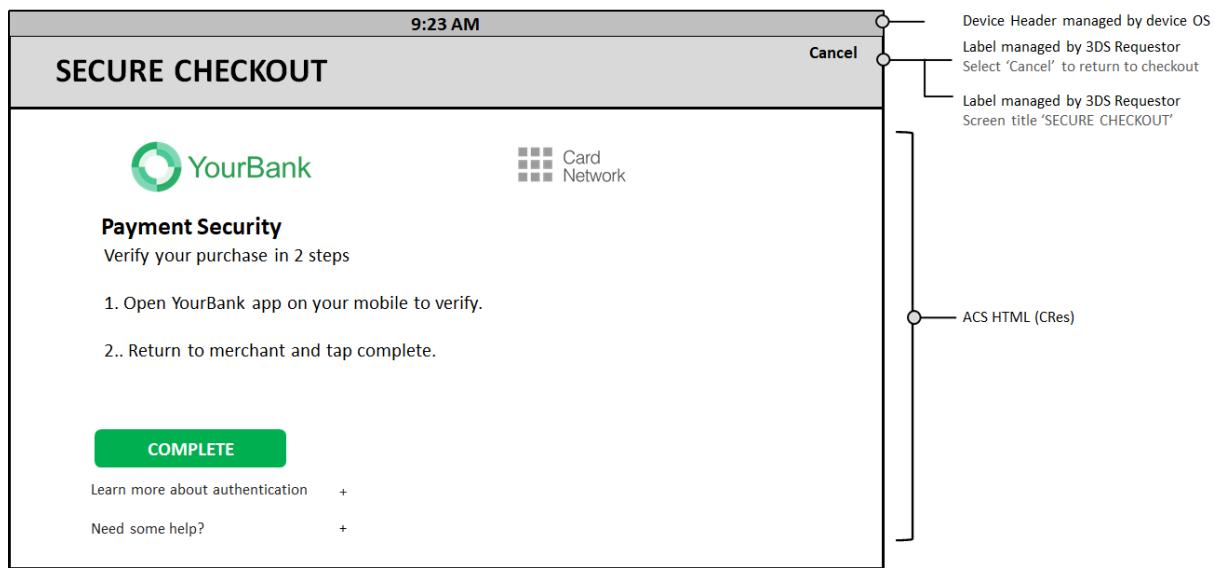


Figure 4.43: Sample OOB HTML UI Template with OOB App URL button—PA—Portrait

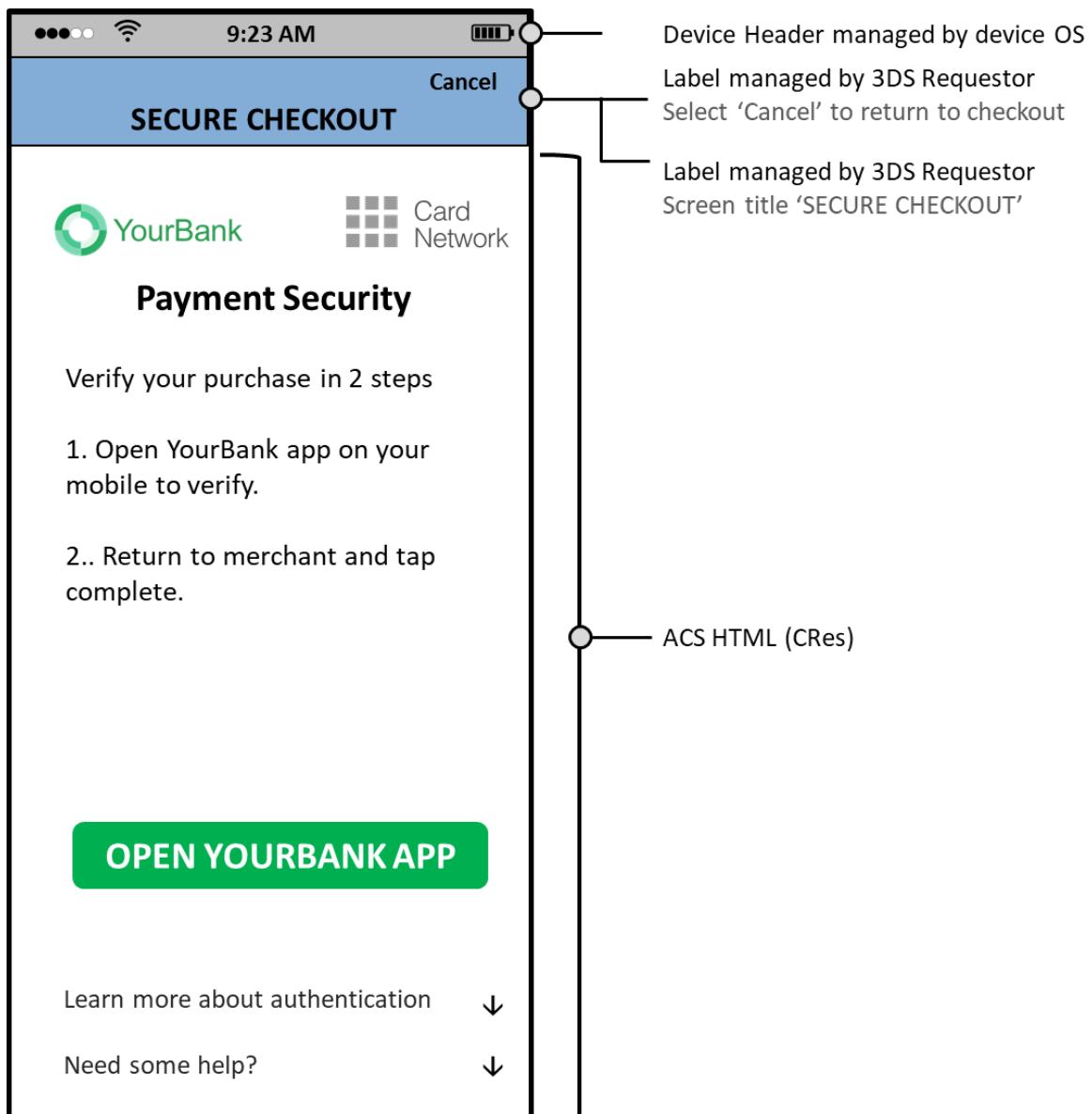


Figure 4.44: Sample OOB HTML UI Template with OOB App URL button—PA—Landscape

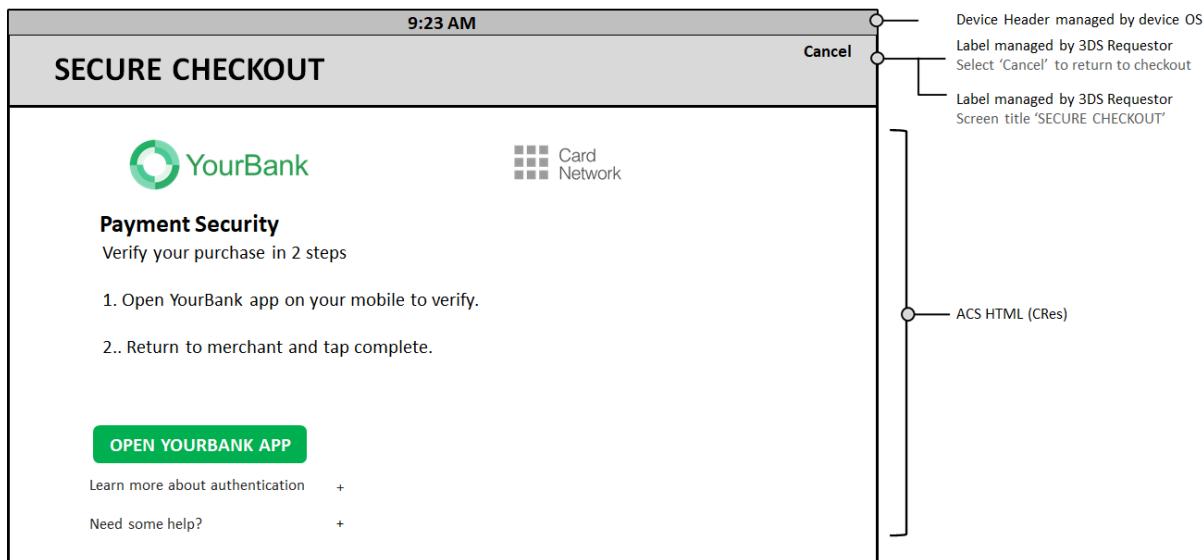


Figure 4.45 and Figure 4.46 provide sample HTML Information UI templates to display instructions to the Cardholder that includes Issuer branding. The Information user interface allows Issuers to display specific information during the challenge, for example to recover from an error situation or for the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

Figure 4.45: Sample Information HTML UI Template—Portrait

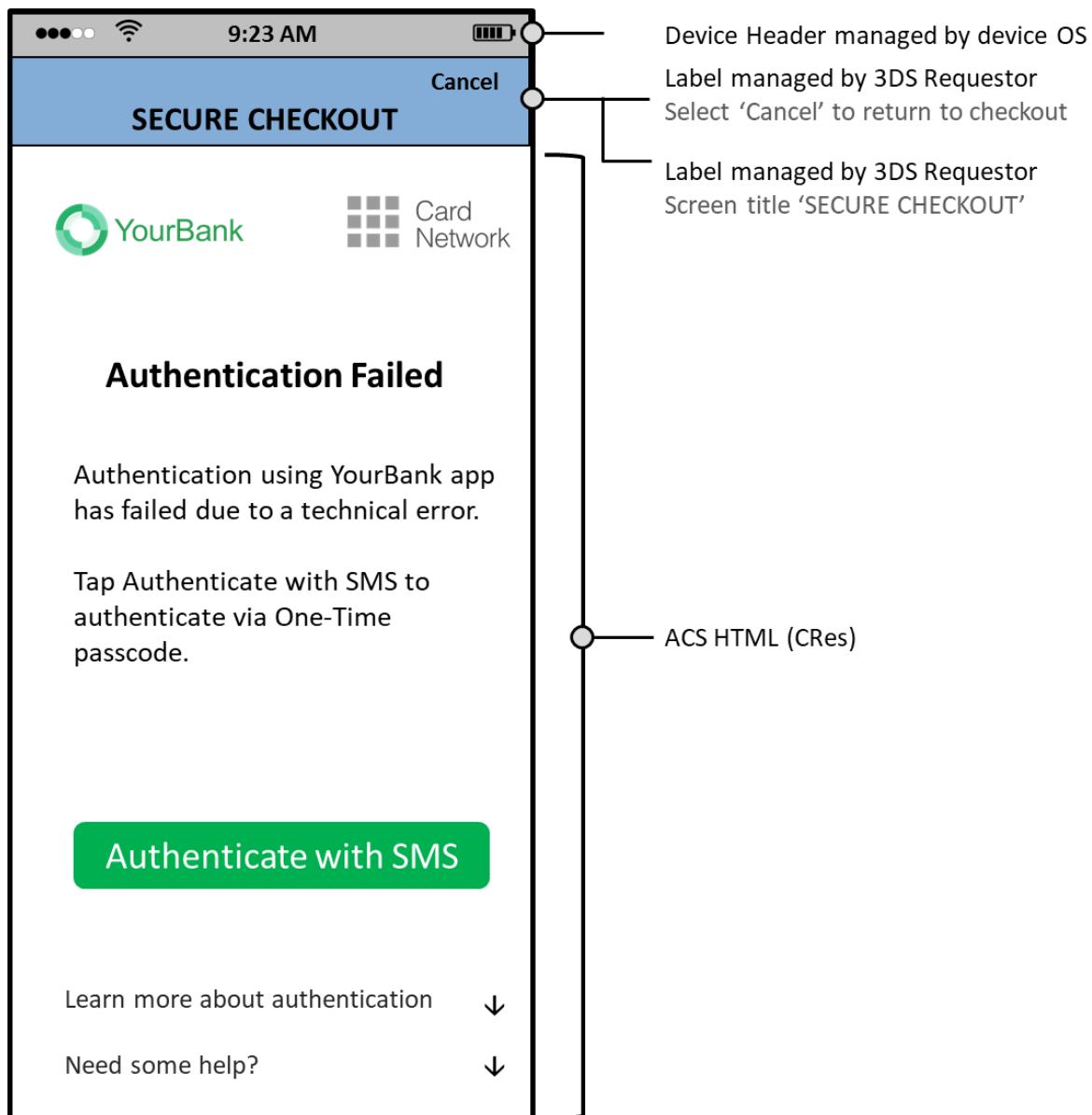
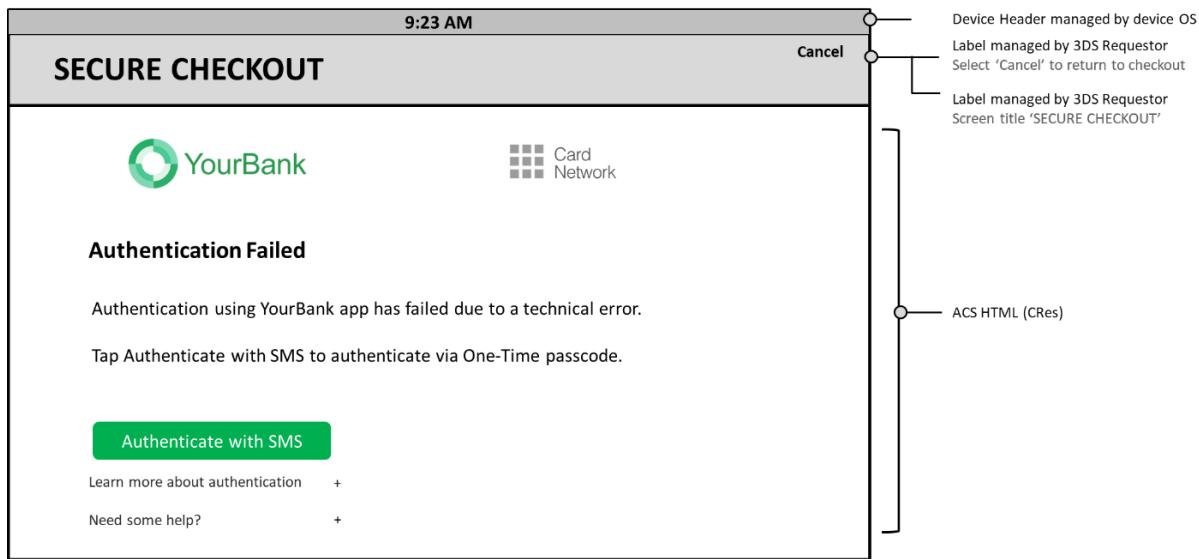


Figure 4.46: Sample Information HTML UI Template—Landscape



4.2.6.1 HTML Other UI Templates

The HTML Other UI Template allows Issuers to perform authentication functionality other than the existing Native data element options standard templates. This option is exclusive to the HTML UI, adheres to the HTML template guidelines and will also be subject to the rules of the specific DS.

Figure 4.47 and Figure 4.48 provide sample HTML Other templates asking the Cardholder to answer questions and confirm an image. There is not an existing data element in the Native format that supports the presentation of an image during authentication, however, the HTML Other will allow for this authentication experience.

Figure 4.47: Sample HTML Other UI Template—PA—Portrait

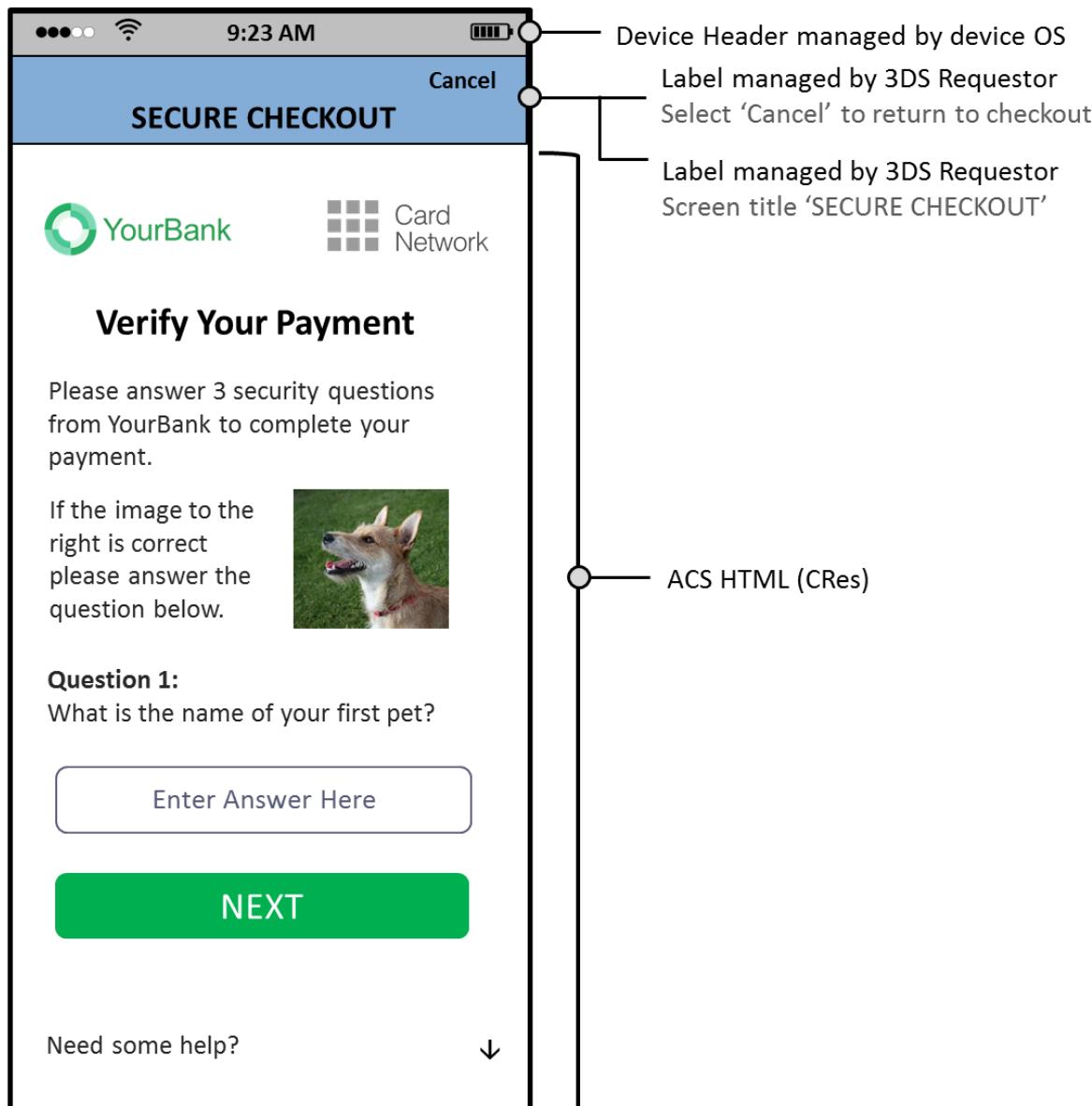
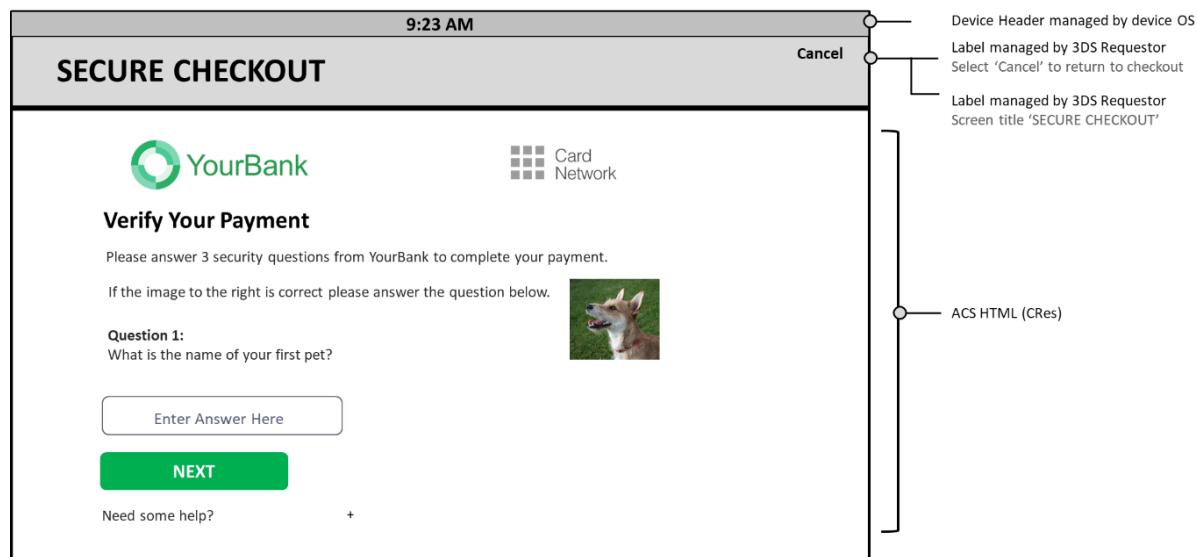


Figure 4.48: Sample HTML Other UI Template—PA—Landscape



4.2.7 HTML Message Exchange Requirements

This section identifies the requirements for an App-based HTML UI.

4.2.7.1 3DS SDK

The 3DS SDK shall:

Seq 4.48 **[Req 159]** After submitting the CReq message to the ACS, display the 3DS App Processing screen until the CRes message is received or timeout exceeded.

4.2.7.2 ACS

The ACS shall:

Seq 4.49 **[Req 160]** Ensure the HTML provided does not contain external references, either navigation attempts or external resource requests.

Seq 4.50 **[Req 161]** Embed CSS and image data to be processed and rendered within the web view entirely within the HTML.

Seq 4.51 **[Req 162]** Provide a fully formed HTML document, including CSS and logos, following responsive design principles.

Seq 4.52 **[Req 163]** Embed all resources in the ACS-provided HTML and not fetched via external URLs.

Seq 4.53 **[Req 164]** Include in the ACS HTML an action which triggers a location change to a specified (HTTPS://EMV3DS/challenge) URL upon the Cardholder completing data input and pressing Submit, for ACS UI Type = 05, and only if manual app switching is used for OOB for ACS UI Type = 06.

Note: The ACS uses the location setting technique described above to communicate back to itself through the secure CReq/CRes channel.

Seq 4.54 **[Req 165]** Not bypass the 3DS SDK or connect back to itself directly.

As defined via the requirements in Section 3.1, the ACS will then use the secure CReq/CRes channel through the 3DS SDK for communication with the Consumer Device.

The ACS will continue a Challenge message cycle until it is complete. Whether the cycle is completed is communicated to the 3DS SDK via the Challenge Completion Indicator data element in the CRes message.

The CRes message sent to the 3DS SDK will include the ACS HTML data element to render the screen as designed by the Issuer/ACS.

4.2.7.3 3DS SDK

The 3DS SDK shall:

- Seq 4.55 **[Req 166]** Extract the HTML data from the ACS HTML data element and display the requested content upon receiving the CRes message from the ACS.
- Seq 4.56 **[Req 167]** Intercept and block any requests by the web view to fetch external resources.
- Seq 4.57 **[Req 168]** Build a view that includes the 3DS Requestor header and place at the top of the view containing the ACS HTML as specified in the HTML UI templates.
- Seq 4.58 **[Req 169]** Use a web view to display the UI to the Cardholder. (WebView, View Controller, etc.).
- Seq 4.59 **[Req 170]** Process Cardholder actions, for example the Cancel action.
- Seq 4.60 **[Req 171]** Return control to the 3DS Requestor App when the Cancel action is selected.

On HTML submit, the Cardholder's response is returned as a parameter string, the form data is passed to the web view instance by triggering a location change to a specified URL ([HTTPS://EMV3DS/challenge](https://EMV3DS/challenge)) with the challenge responses appended to the URL.
- Seq 4.61 **[Req 413]** Monitor the URL changes, to retrieve the Cardholder response as query parameters from the URL ([HTTPS://EMV3DS/challenge](https://EMV3DS/challenge)) and return a parameter string (HTML Action = GET) containing the Cardholder data input.
- Seq 4.62 **[Req 393]** Pass the received data input, unchanged, to the ACS in the Challenge HTML Data Entry data element of the CReq message. The 3DS SDK shall not modify or reformat this received data input.

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

Example Cardholder Response

[HTTPS://EMV3DS/challenge?response=1234&submit=verify](https://EMV3DS/challenge?response=1234&submit=verify), the 3DS SDK returns the cardholder data input "challengeHTMLDataEntry" :
"response=1234&submit=verify"

4.3 Browser-based User Interface Overview

The Browser UI provides Cardholders with an Issuer-specific, consistent Browser-based experience across 3DS Requestors. When a challenge is necessary, the ACS provides HTML to the Browser for display to the Cardholder using the 3-D Secure UI design guidelines. Detailed requirements are described in the documentation provided by each DS.

Note: Browser Flows will function on all devices that support Browser display.

In the Browser UI, no header information is necessary as the UI appears within the Browser window, either within a Lightbox or Inline. The 3DS Requestor/3DS Integrator is responsible for providing the size of the iframe as well as the placement in the 3DS Requestor website.

4.3.1 Processing Screen Requirements

The Browser Processing screen is displayed at the start of all 3-D Secure Browser-based transaction flows.

4.3.1.1 3DS Requestor Website

The 3DS Requestor website shall:

- Seq 4.63 **[Req 172]** Create a Processing screen with a Processing Graphic (for example, a progress bar or a spinning wheel) for display during the AReq/ARes message cycle.

Note: The Processing screen is displayed by the 3DS Requestor website during AReq message processing.

- Seq 4.64 **[Req 173]** Display the Processing screen that conveys to the Cardholder that processing is occurring (Refer to Figure 4.50 and Figure 4.51 for examples).

- Seq 4.65 **[Req 174]** Include the DS logo for display with or without a white box at the centre of the screen unless specifically requested not to include.

- Seq 4.66 **[Req 175]** Not include any other design element or text in the Processing screen.

- Seq 4.67 **[Req 176]** Display the Processing screen for a minimum of two seconds.

4.3.1.2 ACS

The ACS shall:

- Seq 4.68 **[Req 177]** Create and maintain versions of the HTML that correspond to the sizes of the Challenge Window Size data element as defined in Table A.1 and provide the appropriate size in the CRes message based upon the Challenge Window Size that was provided by the 3DS Server in the CReq message.

- Seq 4.69 **[Req 178]** Create a Processing screen without words, text or white box for display during the HTML exchange CReq/CRes message cycle.

- Seq 4.70 **[Req 181]** Not include the DS logo or any other design element in the Processing screen.

- Seq 4.71 **[Req 179]** Display a graphical element (for example, a progress bar or a spinning wheel) within the Challenge/Processing zone that conveys to the consumer that processing is occurring.

- Seq 4.72 **[Req 180]** Include the DS logo in the HTML for display in the Branding Zone unless specifically requested not to include.

- Seq 4.73 **[Req 182]** Display the Processing screen for a minimum of one second.

- Seq 4.74 **[Req 183]** Ensure Browser compatibility, by using a commercial CA that is supported by major Browsers.

4.3.2 Browser Display Requirements

The Browser will display the HTML as provided by the ACS. As such, it is the ACS responsibility to format the HTML to best display on the Consumer Device.

4.3.2.1 ACS

The ACS shall for the CReq/CRes message exchange:

- Seq 4.75 **[Req 380]** Create HTML with form elements in the applicable zones as outlined in Figure 4.1 and Figure 4.2 to support both portrait and landscape UI templates. The format is outlined in the UI templates in Section 4.3.3.

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

- Seq 4.76 **[Req 381]** Expandable Information Label for display as a graphical control element that can be expanded (for example, an accordion) in the Information zone.
- Seq 4.77 **[Req 382]** Expandable Information Text for display in the Information zone only when the Cardholder selects the Expandable Information Label.
- Seq 4.78 **[Req 383]** Why Information Label for display as a graphical control element that can be expanded (for example, an accordion) in the Information zone.
- Seq 4.79 **[Req 384]** Why Information Text for display in the Information zone when the Cardholder selects the Why Information Label.

4.3.3 Browser UI Templates

The figures provided in this section depict examples of the Issuer content and format, as well as the 3DS Requestor website placement.

Figure 4.49 depicts the consistency between the App-based HTML and the Browser UI.

Figure 4.49: App-based HTML and Browser UI Comparison

HTML App-based Template	Browser Template
<p>SECURE CHECKOUT</p> <p>YourBank Card Network</p> <p>Payment Security</p> <p>Please answer 3 security questions from YourBank to complete your payment.</p> <p>Select all that apply</p> <p>Question 2: What cities have you lived in?</p> <p><input checked="" type="checkbox"/> Chicago, Illinois <input type="checkbox"/> Portland, Oregon <input checked="" type="checkbox"/> Dallas, Texas <input type="checkbox"/> St Louis, Missouri</p> <p>NEXT</p> <p>Learn more about authentication ↓</p>	<p>Cancel</p> <p>YourBank Card Network</p> <p>Payment Security</p> <p>Please answer 3 security questions from YourBank to complete your payment.</p> <p>Select all that apply</p> <p>Question 2: What cities have you lived in?</p> <p><input checked="" type="checkbox"/> Chicago, Illinois <input type="checkbox"/> Portland, Oregon <input checked="" type="checkbox"/> Dallas, Texas <input type="checkbox"/> St Louis, Missouri</p> <p>NEXT</p> <p>Learn more about authentication ↓</p>

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

Figure 4.50: Sample Browser Lightbox Processing Screen without White Box

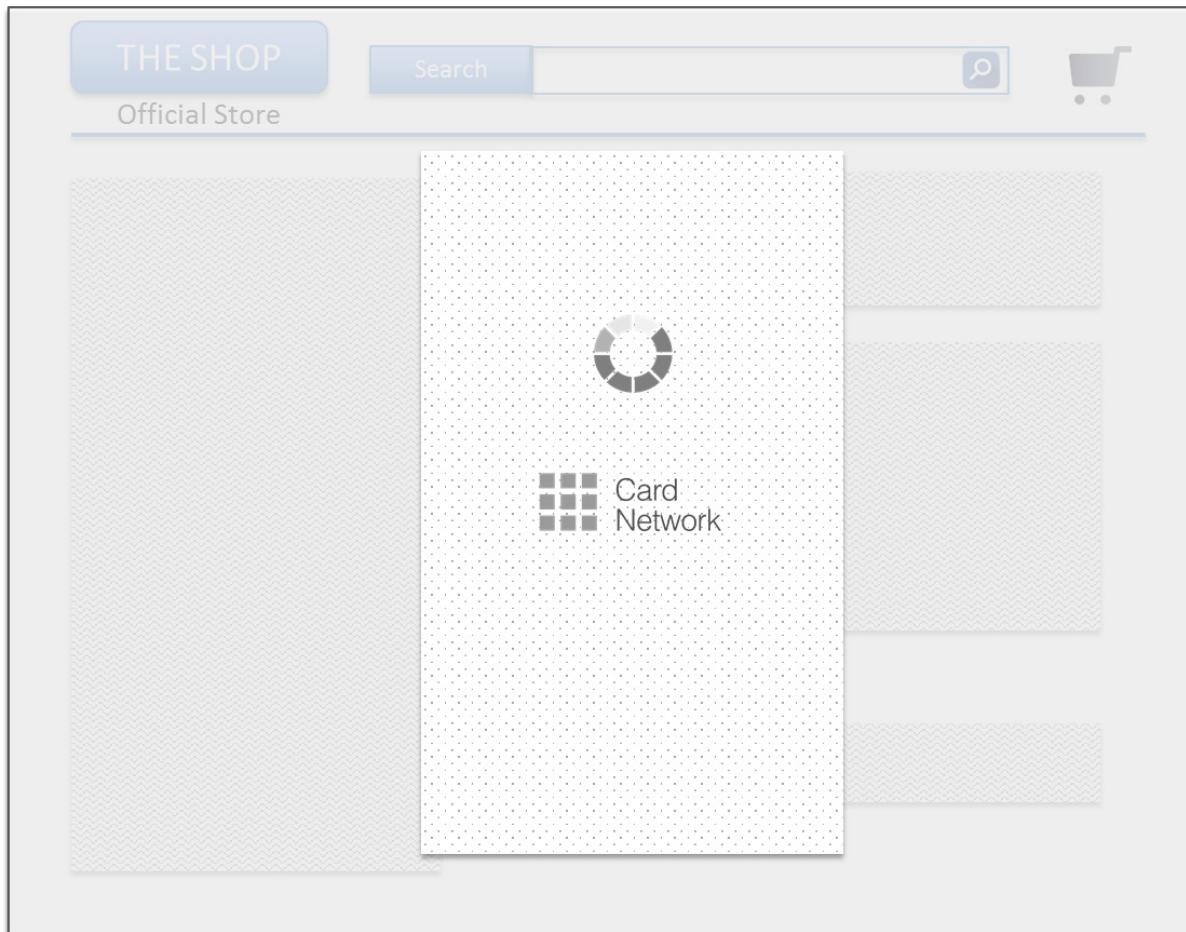


Figure 4.51 depicts a sample Inline Browser Processing screen with a white box.

Figure 4.51: Sample Inline Browser Processing Screen with White Box

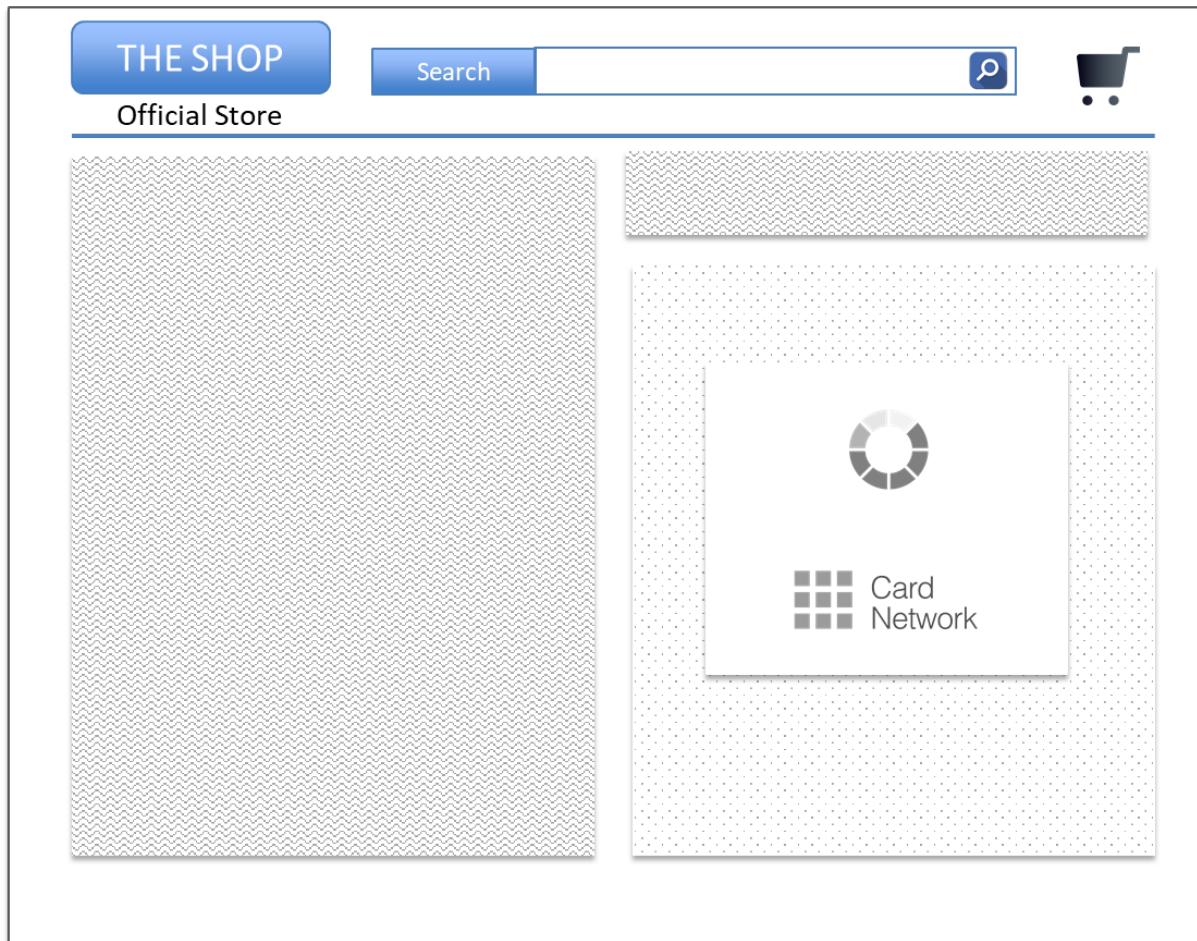


Figure 4.52 depicts a sample Browser with Lightbox UI.

Figure 4.52: Sample Browser with Lightbox UI—PA

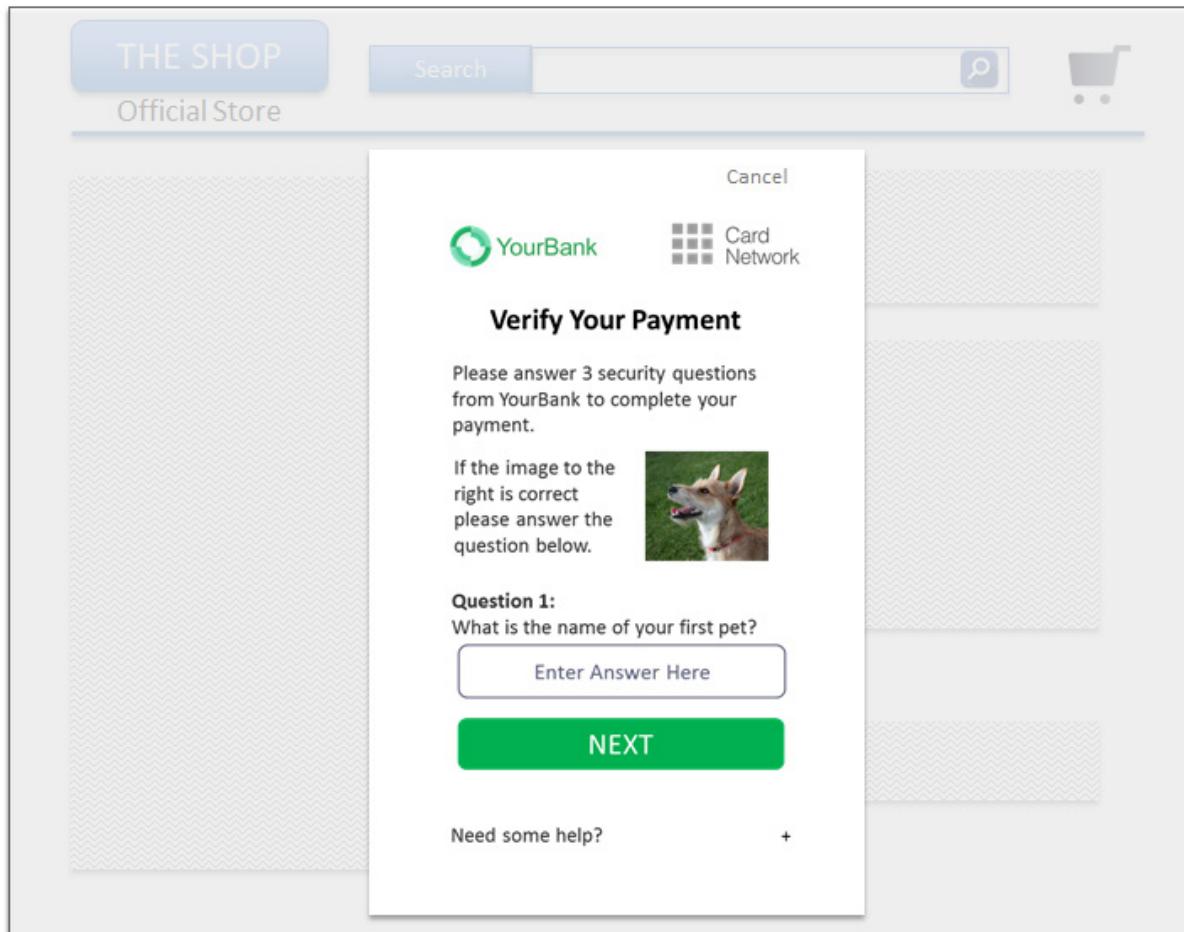
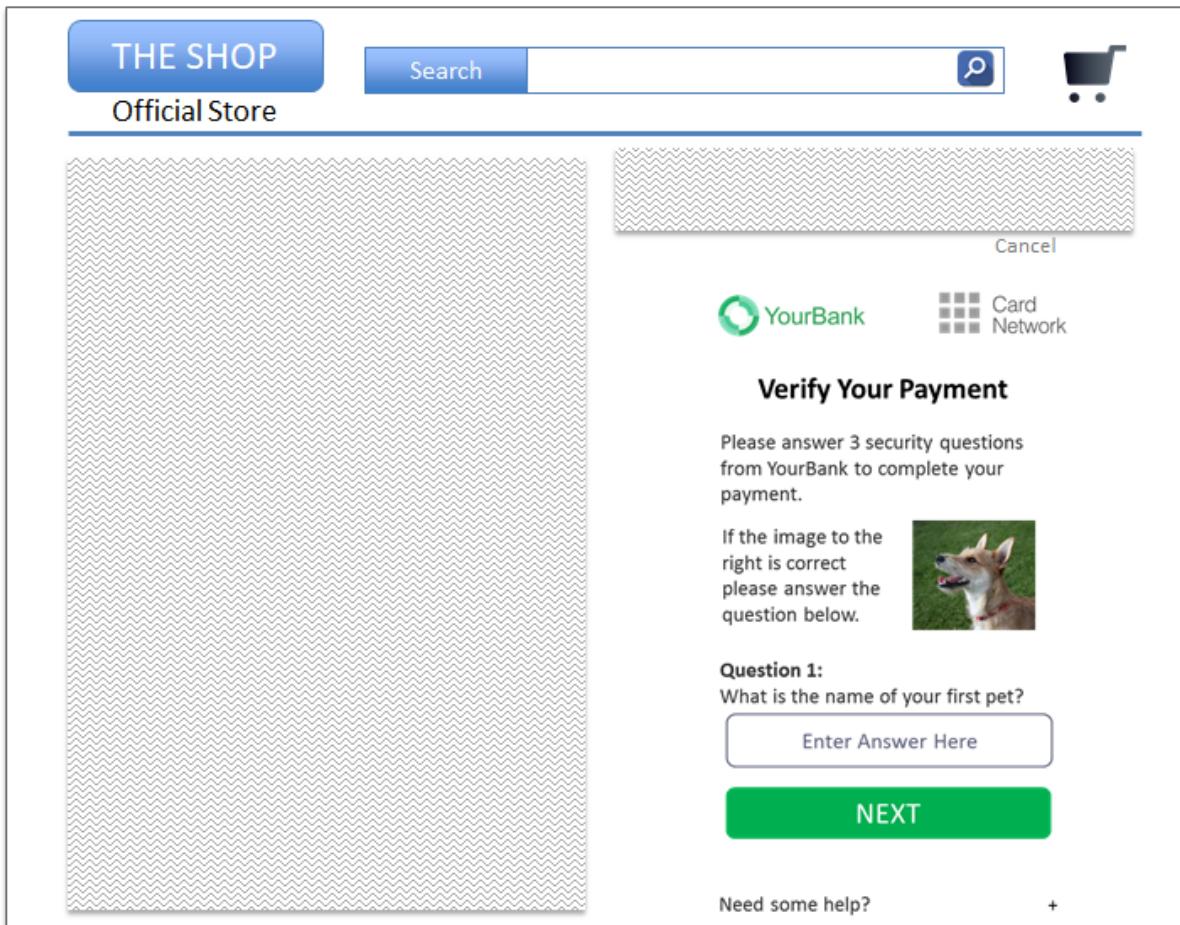


Figure 4.53 depicts a sample Browser with Inline UI.

Figure 4.53: Sample Browser with Inline UI—PA



Note: For Browser-based Decoupled Authentication transactions, the 3DS Requestor website displays the Processing Screen followed by a display of the Cardholder Information Text within the 3DS Requestor's checkout page (this is the same as App-based Decoupled transactions, as depicted in Figure 4.13).

4.4 3RI Considerations

The two types of 3RI transactions have different UI considerations. The first type mainly concerns recurring transactions where the 3DS Requestor wants to verify that a subscription user still has a valid form of payment. As the cardholder is not present, there is no user interface presented for this type of 3RI transaction.

The second type of 3RI transaction is when the 3DS Requestor requests Decoupled Authentication as a method to authenticate the Cardholder, for example, a MOTO transaction. The user interface provided by the ACS for Decoupled Authentication is outside the scope of the EMV 3-D Secure protocol. The 3DS Requestor can display a Processing Screen during AReq/ARes message processing and convey the Cardholder Information Text element if provided by the ACS in the ARes message. For a MOTO transaction, the 3DS Requestor's customer service representative, can convey the Cardholder Information Text verbally for transactions conducted via a land line or feature phone.

5 EMV 3-D Secure Message Handling Requirements

This chapter details the functions of the 3-D Secure message handling including connection establishment, message parsing and validation, as well as message and error handling. The 3-D Secure messages across all flows utilise open Internet Standards to improve interoperability across domains, and those Standards are referenced in this section of the specification.

5.1 General Message Handling

5.1.1 HTTP POST

To ensure interoperability, all 3DS components shall follow these methods for HTTP POST:

- Seq 5.1 **[Req 184]** All 3-D Secure message requests shall be sent via HTTP POST as defined in RFC 7231 over a secured connection as defined in Section 6.1.
- Seq 5.2 **[Req 185]** Messages exchanged between 3-D Secure components shall be in the JSON data interchange format as defined in RFC 7159, or JWE/JWS object format as defined in RFC 7516/RFC 7515.
- Seq 5.3 **[Req 186]** The body of the HTTP message shall contain the JSON message properly formatted utilising the JSON required UTF-8 character set as defined in RFC 7159, or JWE/JWS object format as defined in RFC 7516/RFC 7515. To maximise the message content, it is recommended to remove any whitespace (space, carriage return, line feed etc.) characters outside of quoted strings of the JSON data.
- Seq 5.4 **[Req 187]** Hypertext Transfer Protocol—HTTP/1.1 or greater shall be utilised for connectivity. Refer to RFC 2616 for detail on HTTP/1.1.
- Seq 5.5 **[Req 188]** If chunked transfer coding is not used, the Content-Length: header shall be present (and set to the length of the message body). Refer to RFC 2616 for detail on HTTP/1.1 chunked transfer coding.
- Seq 5.6 **[Req 189]** Responses shall be formatted as defined in Section 5.1.2 (including the formatting and Content-Type header) and sent in the reply to the HTTP POST.
- Seq 5.7 **[Req 313]** Cryptographic keys exchanged between 3-D Secure components shall be in the JWK object format as defined in RFC 7517.

5.1.2 HTTP Header—Content-Type

The HTTP headers in the 3-D Secure messaging convey the content of the message body for receiving components to properly process the content.

3-D Secure utilises the HTTP headers, and specifically, the Content-Type Header field defined in RFC 2616.

- Seq 5.8 **[Req 190]** The HTTP headers shall contain the Content-Type Header: application/JSON; and include charset of UTF-8 for the following messages:
- AReq/ARes

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

For example, Content-Type: application/JSON; charset = UTF-8

Seq 5.9 **[Req 191]** The Content-Type Header requirements for CReq/CRes are:

- For App-based CReq/CRes, the HTTP headers shall contain the Content-Type Header: application/jose; and include charset of UTF-8.

For example, Content-Type: application/jose; charset = UTF-8

- For Browser-based CReq the HTTP headers shall contain the Content-Type Header: application/x-www-form-urlencoded.

For example, Content-Type: application/x-www-form-urlencoded

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

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

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

Seq 5.10 **[Req 468]** For the AReq, CReq, RReq, OReq or PReq messages, the 3DS component sending the message shall include its own Transaction ID using the X-Request-ID in the HTTP header, as defined in Table A.29.

Seq 5.11 **[Req 469]** For the ARes, CRes, RRes, ORes or PRes messages, the 3DS component sending the message shall include its own Transaction ID using X-Response-ID and the X-Request-ID received in the Request message in the HTTP header, as defined in Table A.29.

5.1.3 Base64/Base64url Encoding

Base64 and Base64url encoding is used throughout the 3-D Secure specification to provide consistency in formatting the content of certain message elements as defined in Annex A.

Seq 5.12 **[Req 192]** The methods of encoding shall follow:

- IETF RFC 2045 for Base64
- IETF RFC 7515 for Base64url

Seq 5.13 **[Req 193]** Base64 decoding software shall ignore any white space (such as carriage returns or line ends) within Base64-encoded data and shall not treat the presence of such characters as an error.

5.1.4 Protocol and Message Version Numbers

Note: Protocol and Message Version Numbers are in the format: major.minor.patch (for example, 2.3.0).

Seq 5.14 **[Req 195]** Any Message Version Number not indicated as active in *EMV Specification Bulletin 255* shall be returned as an error. The 3-D Secure component shall return an Error Message with the applicable Error Component and an Error Code = 102.

Seq 5.15 **[Req 320]** The Message Version Number is set by the 3-D Secure component initiating the 3DS message and the 3-D Secure components shall validate that the Message Version Number remains unchanged throughout the 3-D Secure transaction. The 3-D Secure component that identifies a validation error shall return an Error Message with the applicable Error Component and Error Code = 203.

[Req 311] 3DS components shall support all lower active protocol versions (Protocol Version Status set to Active in *EMV Specification Bulletin 255*). 3DS components shall support the latest patch for their current version. 3-D Secure messages containing an active Message Version Number supported by the 3-D Secure component shall be processed according to the requirements of the specified protocol version (See *EMV Specification Bulletin 255*).

5.1.5 Data Version Numbers

Seq 5.16 **[Req 396]** The 3DS SDK shall support the latest Data Version of the 3DS SDK Device Information.

Seq 5.17 **[Req 397]** The ACS shall support all active Data Versions of the 3DS SDK Device Information.

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

5.1.6 Message Parsing

The recipient of a 3-D Secure message validates the message to ensure that it can be correctly processed.

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

Seq 5.18 **[Req 196]** Message meets applicable requirements as defined in Annex A.

Seq 5.19 **[Req 197]** Context of the transaction is valid for the receiving component based on the Message Type field.

Seq 5.20 **[Req 198]** 3-D Secure message is properly formatted as a JSON message as defined in RFC 7159 or JWE/JWS object format as defined in RFC 7516/7515.

Seq 5.21 **[Req 199]** Message Type field is valid for the receiving component.

The receiving component receives the following message types:

Seq 5.22 **[Req 200]** The 3DS SDK shall only accept the following messages: CRes or Error Message. Any other message type shall be treated as an error.

Seq 5.23 **[Req 201]** The 3DS Server shall only accept the following messages: ARes, RReq, PRes OReq or Error Message. Any other message types shall be treated as an error.

Seq 5.24 **[Req 202]** The DS shall only accept the following messages: AReq, ARes, RReq, RRes, PReq, ORes or Error Message. Any other message types shall be treated as an error.

Seq 5.25 **[Req 203]** The ACS shall only accept the following messages: AReq, CReq, RRes, OReq or Error Message. Any other message types shall be treated as an error.

Seq 5.26 **[Req 430]** If the 3DS Server receives more than one RReq message during a transaction, then it shall return Error Code = 312.

- Seq 5.27 **[Req 431]** If the 3DS Server receives an RReq message and the Transaction Status does not = C or D or S in the corresponding ARes message, then it shall return Error Code = 313.
- Seq 5.28 **[Req 432]** If the DS receives more than one RReq message during a transaction, then it shall return Error Code = 312.
- Seq 5.29 **[Req 433]** If the DS receives an RReq message and the Transaction Status does not = C or D or S in the corresponding ARes message, then it shall return Error Code = 313.

5.1.7 Message Content Validation

During the validation of the message, the following requirements apply:

- Seq 5.30 **[Req 204]** All Required fields for the Message Type received shall be present as defined in Table A.1.
- Seq 5.31 **[Req 205]** All Conditional fields for the Message Type received shall be present when the source component determines the conditions for presence are met as defined in Table A.1.
- Seq 5.32 **[Req 206]** Required, Conditional, and Optional fields contained in the Message Type shall meet formatting and length criteria as specified in Table A.1.
- Seq 5.33 **[Req 309]** Unless explicitly noted, if a conditionally optional or optional field is sent as empty or null, the receiving component shall return an Error Message (as defined in Section A.9) with the applicable Error Component and Error Code = 203.

For example:

The DS receives an ARes message from the ACS with an empty conditionally Optional data element that is specified in Table A.1 for the Message Type, Device Channel and Message Category but the condition is not met. Such as, acsChallengeMandated = "" and transStatus = Y. The DS validates the ARes message content and returns an error to the ACS and can return an ARes message or Error to the 3DS Server.

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

- Seq 5.34 **[Req 207]** To support future versions of the protocol, implementations shall not use (or configure) JSON content parsers that validate strictly. If the message is syntactically correct and the validation criteria for the Message Type are met as defined in Table A.1, then the message shall be considered valid.
- Seq 5.35 **[Req 208]** If the content validation for the Message Type received does not pass validation criteria as defined in Table A.1, then the message shall be treated as an error.
- Seq 5.36 **[Req 209]** If there are additional data elements received that are not specified for the Message Type, Device Channel and Message Category but the message otherwise passes validation, the message shall be considered valid.
For the additional data elements received (with the exception of data extensions), the receiving 3DS component shall EITHER:
 - Ignore the additional data elements and not send them to the next 3DS component in the flowOR

- Check the format of the additional data elements:
 - If the format is correct, ignore the additional data elements and do not send them to the next 3DS component in the flow.
 - If the format is incorrect, the receiving 3DS component responds with an error message to the sending 3DS component.

For example:

- The DS receives an AReq message from the 3DS Server with additional data elements that are not specified in Table A.1 for the AReq Message Type, Device Channel and Message Category and the DS validates the AReq content and drops the additional elements when sending the AReq message to the ACS.

OR
- If the additional data elements in the AReq message do not pass format checking, the DS can respond with an error message to the 3DS Server.

Seq 5.37 **[Req 210]** All 3-D Secure components shall ignore unrecognised non-critical extension name/value pairs (that is, any extension that does not have a criticality attribute with a value = true) and pass them.

Seq 5.38 **[Req 434]** If the value of a data element is in the range of “Reserved for EMVCo future use”, all 3DS components shall return an Error Message (as defined in Section A.9) with the applicable Error Component and Error Code = 207.

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

Content Validation Requirements:

Seq 5.39 **[Req 212]** 3-D Secure components shall ensure response message data elements as defined in Table A.1, including Transaction IDs and Reference Numbers match that of the corresponding request message.

Seq 5.40 **[Req 213]** Upon finding any message data elements that do not pass format validation, the validating component shall generate a response message having the Error Detail include the data element name(s) of the incorrect elements populated. Refer to Table A.4 for detailed information.

Note: The AReq message will have additional message content validation requirements based on the content of the data elements.

5.2 Partial System Outages

A 3-D Secure component may experience system failures that effectively render the component inoperable.

Seq 5.41 **[Req 215]** When system failures are detected, the system shall stop accepting requests until the failure has been corrected and an Error Message as defined in Section A.9 with the applicable Error Component and an Error Code = 403 or 404 shall be sent for any outstanding requests.

5.3 3-D Secure Component Availability

All 3-D Secure components are recommended to be architected with high availability as a key factor in the software, system and infrastructure design to maintain the integrity of the entire 3-D Secure ecosystem. Additional recommendations are:

- The availability of any one 3-D Secure component should not rely on any other component to handle message routing or load balancing.
- While 3-D Secure components may optionally store multiple URLs for routing purposes, they should not be leveraged as the only high availability solution.

Note: Details about high availability best practices are outside the scope of EMV 3-D Secure.

5.4 Error Codes

- Seq 5.42 **[Req 216]** All 3-D Secure components shall accept any value in the Error Code field.
- Seq 5.43 **[Req 217]** If the list of Error Code values in Table A.4 does not contain an entry that matches a condition detected by a 3-D Secure component, a reasonably close match shall be used.
- Seq 5.44 **[Req 218]** For Error Codes 201, 203, 204 and 304, the 3-D Secure component that identifies the error shall include the name(s) of the erroneous data element(s) in the Error Detail.

5.5 Timeouts

This section provides requirements for transaction timeouts and read timeouts.

5.5.1 Transaction Timeouts

Transaction Timeouts provide the maximum amount of time that a 3-D Secure transaction should be considered valid. 3-D Secure components are responsible for maintaining transaction timeout values as defined by this specification.

The ACS shall:

- Seq 5.45 **[Req 219]** Maintain the state of a 3-D Secure transaction when the ARes message contains the Transaction Status = C so that the corresponding CReq message can be processed and timeout values can be enforced.
- Seq 5.46 **[Req 220]** Upon sending an ARes message with the Transaction Status = C, set a timeout value of 30 seconds for the receipt of the initial corresponding CReq message from the 3DS SDK or 3DS Requestor.

- Seq 5.47 **[Req 221]** If the transaction reaches the 30-second timeout expiry and an Error Message has not been received from the DS for this transaction, send an RReq message to the DS with Transaction Status = N, Transaction Status Reason = 14 (Transaction timed out at the ACS), and Challenge Cancelation Indicator = 05 (Transaction timed out at the ACS—First CReq message not received). Clear the ephemeral key generated and stored for use in the CReq/CRes message exchange for the current transaction.
- Seq 5.48 **[Req 222]** Upon receiving a CReq message for a transaction that has timed-out, send an Error Message with Error Component = A and Error Code = 402 to the 3DS SDK (for an App-based transaction) or 3DS Requestor (for a Browser-based transaction).

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

- Seq 5.49 **[Req 223]** Set a timeout value of 10 min (or 600 sec) after successfully sending each CRes message to the 3DS SDK.
- Seq 5.50 **[Req 224]** If the timeout expires before receiving the next CReq message from the 3DS SDK, send an RReq message within 60 seconds to the DS to be passed to the 3DS Server with Transaction Status = N, Transaction Status Reason = 14 (Transaction timed out at the ACS), and Challenge Cancelation Indicator = 04 and then clear any ephemeral key generated and stored for use in the CReq/CRes message exchange for this transaction.
- Seq 5.51 **[Req 225]** Upon receiving the CReq message for a transaction that has timed out, send an Error Message with Error Component = A and Error Code = 402 to the 3DS SDK.

For Browser-based transactions, once a transaction has been established with a successful CReq POST to the ACS from the 3DS Requestor, and when the ACS sends the challenge interface to the challenge iframe, the ACS shall:

- Seq 5.52 **[Req 226]** Set a timeout value of 10 min (or 600 seconds) after successfully sending each challenge interface to the challenge iframe.
- Seq 5.53 **[Req 227]** If the timeout expires before Cardholder authentication can complete, send an RReq message within 60 seconds to the DS to be passed to the 3DS Server with the Transaction Status = N and Transaction Status Reason = 14.

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

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

- Seq 5.54 **[Req 344]** Close the challenge iframe by refreshing the parent page and removing the HTML iframe.

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

- Seq 5.55 **[Req 452]** Set a timeout value of 10 minutes (or 600 seconds) after sending the first ARes message.

- Seq 5.56 **[Req 453]** If the timeout expires before the second AReq message is received, send an RReq message within 60 seconds to the DS, to be passed to the 3DS Server, with Transaction Status = N and Transaction Status Reason = 14.

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

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

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

- Seq 5.57 **[Req 454]** Set a timeout value of 9 minutes (or 540 seconds) after successfully providing the SPC data to the 3DS Requestor.
- Seq 5.58 **[Req 455]** If the timeout expires before receiving the Assertion Data from the 3DS Requestor, send the second AReq message to the ACS with SPC Incompletion Indicator = 03.

5.5.2 Read Timeouts

To ensure that 3-D Secure messages are handled across components in a timely manner, all components set read timeouts as appropriate for the implementation (i.e., programming language function, infrastructure components, etc.).

Read timeouts occur while waiting on a transaction response after the connection has been established and the message is sent to the recipient. Connection timeouts occur while making the initial connection (i.e., completing the TCP/IP connection and TLS handshake).

Note: The read timeout values will be defined by each DS implementation and may vary per DS.

5.5.2.1 AReq/ARes Message Timeouts

Read timeouts in the AReq/ARes messages are handled as follows:

- Seq 5.59 **[Req 228]** The 3DS Server and ACS shall set appropriate AReq and ARes message timeout values, as set by DS requirements when communicating with each DS separately.

The 3DS Server:

- Seq 5.60 **[Req 229]** Any failure to complete the initial TCP/IP connection and TLS handshake to the DS shall result in an immediate retry or the 3DS Server shall try an alternate DS (if available). Upon second failure, the 3DS Server shall send an error to the 3DS Requestor to complete the transaction and may send an Error Message with Error Component = S and Error Code = 405 to the DS.
- Seq 5.61 **[Req 231]** Set the read timeout value as defined by the DS receiving the request from the time the TLS handshake has completed and the full AReq message is sent for processing.
- Seq 5.62 **[Req 232]** If the DS has not responded with the ARes message before the 3DS Server read time expiry, the 3DS Server shall close the connection and result in a failed 3-D Secure transaction.

The DS:

- Seq 5.63 **[Req 233]** Any failure to complete the initial TCP/IP connection and TLS handshake to the ACS shall result in an immediate retry or the DS shall try an alternate ACS (if available). Upon second failure, the DS shall send:
- an Error Message with Error Component = D and Error Code = 405 to the 3DS Server to complete the transaction, OR
 - an ARes message to the 3DS Server (as defined in Table B.2) with Transaction Status set to the appropriate response as defined by the specific DS.
- Note: The DS may maintain multiple ACS URLs. If the first URL attempted is not available, then the DS will attempt to connect to one of the alternate URLs.**
- Seq 5.64 **[Req 424]** If all attempts to successfully connect to the ACS fail, then the DS may send an Error Message with Error Component = D and Error Code = 405 to the ACS.
- Seq 5.65 **[Req 234]** The DS shall set the ACS timeout value (as defined in the relevant DS requirements) from the time the TLS handshake has completed and the full AReq message is sent for processing to the ACS.
- Seq 5.66 **[Req 235]** If the DS has not received the ARes message from the ACS before the read timeout expiry, the DS shall send:
- an Error Message with Error Component = D and Error Code = 402 to the 3DS Server to complete the transaction, OR
 - an ARes message to the 3DS Server (as defined in Table B.2) with Transaction Status set to the appropriate response as defined by the specific DS, and then send an Error Message with Error Component = D and Error Code = 402 to the ACS to complete the transaction.

Read timeouts for the CReq/CRes messages are handled as follows:

The 3DS SDK:

- Seq 5.67 **[Req 236]** Any failure to complete the initial connection and TLS handshake to the ACS shall result in an immediate retry. Upon second failure, the 3DS SDK shall report an error to the 3DS Requestor App to complete the transaction.
- Seq 5.68 **[Req 237]** The 3DS SDK shall set a 10 second timeout value from the time the TLS handshake has completed and the full CReq message is sent for processing to the ACS.
- Seq 5.69 **[Req 239]** If the ACS does not respond with the CRes message before the 3DS SDK 10-second read timeout expiry, the 3DS SDK **ends 3-D Secure processing**, passes an Error Message to the ACS with Error Component = C and Error Code = 402 and passes an error message to the calling app, ending the 3-D Secure transaction.
- Seq 5.70 **[Req 312]** The 3DS SDK shall set the SDK Maximum Timeout value from the time the TLS handshake has completed and the first CReq message is sent for processing to the ACS.
- If the ACS does not respond with the final CRes message before the SDK Maximum Timeout expiry, the 3DS SDK **ends 3-D Secure processing**, passes an Error Message to the ACS with Error Component = C and Error Code = 402 and passes an error message to the calling app, ending the 3-D Secure transaction.

The SDK Maximum Timeout shall not have a value less than five minutes.

5.5.2.2 RReq/RRes Message Timeouts

Read timeouts for the RReq/RRes messages are handled as follows:

The ACS:

- Seq 5.71 **[Req 240]** Any failure to complete the initial connection and TLS handshake to the DS shall result in an immediate retry. Upon second failure, the ACS will wait 10 seconds and retry to connect to the DS until the message is delivered with Transaction Status = U.
- Seq 5.72 **[Req 241]** The ACS shall set a 5-second timeout value from the time the TLS handshake has completed and the full RReq message is sent for processing to the DS.
- Seq 5.73 **[Req 242]** If the DS has not responded with the RRes message or an Error message before the read timeout expiry, the ACS shall return to the DS an Error Message (as defined in Section A.5.5) with Error Component = A and Error Code = 402. The default timeout value is 5 seconds. However, a DS may specify a higher alternative value.

The DS:

- Seq 5.74 **[Req 243]** Any failure to complete the initial connection and TLS handshake to the 3DS Server shall result in an immediate retry. Upon second failure, the DS shall send an Error Message with Error Component = D and Error Code = 405 to the ACS to complete the transaction and may send an Error Message with Error Component = D and Error Code = 405 to the 3DS Server and **ends 3-D Secure processing**.
- Seq 5.75 **[Req 244]** Shall set a timeout value from the time the TLS handshake has completed and the full RReq message is sent for processing to the 3DS Server URL. The default timeout value is 3 seconds; however, a DS may specify a higher alternative value.

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

- Seq 5.76 **[Req 245]** If the 3DS Server has not sent the RRes message before the 3-second (or DS alternative value) read timeout expiry, the DS shall send an Error Message with Error Component = D and Error Code = 402 to the ACS to complete the transaction and may send an Error Message with Error Component = D and Error Code = 402 to the 3DS Server and **ends 3-D Secure processing**.

5.6 PReq/PRes Message Handling Requirements

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

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

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

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

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

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

If the Serial Number has not changed, the DS does not provide back the Card Range Data element but includes the Serial Number in the PRes message.

The 3DS Server shall:

Seq 5.77 **[Req 246]** Call each registered DS for:

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

Seq 5.78 **[Req 425]** Request the Card Range Data under a compressed format by adding “Accept-Encoding: gzip” to the HTTP request.

Seq 5.79 **[Req 456]** Support Range Requests, as defined by RFC 7233, if the Card Range Data Download Indicator is present in the PReq message.

Seq 5.80 **[Req 247]** Send the PReq message via a secure link with the DS established as defined in Section 6.1.2.1.

Seq 5.81 **[Req 248]** Immediately retry a connection upon any failure to complete the initial TCP/IP connection and TLS handshake to the DS.

Seq 5.82 **[Req 249]** Upon the second failure to complete the TCP/IP connection and TLS handshake to the DS, end the transaction, and periodically retry within the 24-hour window at 60-second intervals until the transaction completes successfully.

The DS shall:

Seq 5.83 **[Req 428]** If the DS receives a request without the “Accept-Encoding: gzip” in the header of a 3DS Server HTTP request, not return an Error Message and return the data in an uncompressed format.

Seq 5.84 **[Req 303]** Receive and validate the PReq message, as defined in Table B.6:

- Return to the 3DS Server an Error Message (as defined in Section A.9) with Error Component = D; AND
 - Error Code = 203, if any data element present fails validation
 - Error Code = 201, if any required data elements are missing

- Error Code = 307, if the Serial Number is invalid
 - Error Code = 103, if the 3DS Server submits more than one request for Card Range Data within one hour.
- Seq 5.85 **[Req 457]** Support Range Requests, as defined by RFC 7233, for the Card Range Data File download.
- Seq 5.86 **[Req 458]** Ensure that there is no overlap or conflict in the card ranges contained in the Card Range Data (Start and End of the card range).
- Seq 5.87 **[Req 459]** Provide in the Card Range Data data element only information about card ranges that are participating in EMV 3-D Secure and are registered with the DS that is responding to the request.
- Seq 5.88 **[Req 460]** Prepare the Card Range Data File if supported by the DS and if the Card Range Data Download Indicator was present in the PReq message. The Card Range Data File contains the entire list of participating card ranges, e.g. the JSON object Card Range Data; {"cardRangeData": [...]}.
- Seq 5.89 **[Req 426]** Prepare the PRes message and send a response with either:
- A compressed response body and a Content-Encoding header that specifies that gzip encoding was used ("Content-Encoding: gzip"), OR
 - An uncompressed response body. The DS shall use the uncompressed response format when it provides the Card Range Data File URL in the PRes message.
- Seq 5.90 **[Req 250]** Send the PRes message containing the DS card range information as defined in the Card Range Data data element defined in Table A.1 or the Card Range Data File URL, if supported.
- If the PReq message does not include a Serial Number, or if the DS does not support partial cache update, the DS PRes message response shall contain the entire list of participating card ranges in the Card Range Data using only Action Indicator = A.

The 3DS Server shall:

- Seq 5.91 **[Req 304]** Receive and validate the PRes message as defined in Table B.7:
- If any data element present fails validation, the 3DS Server:
 - Returns to the DS an Error Message (as defined in Section A.9) with Error Component = S and Error Code = 203.
 - If any required data elements are missing, the 3DS Server:
 - Returns to the DS an Error Message (as defined in Section A.9) with Error Component = S and Error Code = 201.

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

- Seq 5.92 **[Req 385]** Update the cache information for each Card Range Data according to the Action Indicator.
- If there is an error in the Card Range Data, then:

- For a Card Range Data overlap, return to the DS an Error Message (as defined in Section A.9) with Error Component = S and Error Code = 205.
- For an Action Indicator error, return to the DS an Error Message (as defined in Section A.9) with Error Component = S and Error Code = 206.
- Discard all updates contained in the Card Range Data, and use previously stored cache information or, alternatively, ignore all existing cache information.
- If the PRes message does not include a Serial Number, replace all existing Card Range Data for the DS using Action Indicator = A for all card ranges returned (i.e., the Action Indicator is ignored in the PRes message).

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

5.7 App/SDK-based Message Handling

The App-based flows have specific requirements for security and functionality that differ from the Browser-based flows. The 3DS SDK is the main component of the 3-D Secure integration into a 3DS Requestor App.

The 3DS SDK shall be developed adhering to the applicable EMV 3DS SDK specification requirements and APIs.

The 3DS SDK has two key functions:

- Provide all data as specified in the *EMV 3-D Secure SDK—Device Information* to be sent through the 3DS Requestor Environment to the 3DS Server and on to the DS and ACS.
- Provide the user interface (UI) and CReq/CRes message handling for the Challenge Flow.

5.7.1 App-based CReq/CRes Message Handling

The CReq/CRes messages are used during the Cardholder authentication and are a direct communication between the Consumer Device (via the 3DS Client) and the ACS.

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

Upon receipt of the ARes message data from the 3DS Requestor App, the 3DS SDK validates the JSON Web Signature (JWS) used to sign the ephemeral keys and the ACS URL, generates the JSON Web Encryption (JWE) object containing the CReq message, and sends the JWE object to the ACS URL received from the 3DS Server:

Seq 5.93 **[Req 253]** The 3DS SDK shall verify the JWS signature found in the ACS Signed Content data element of the ARes message received from the 3DS Server as defined in Section 6.2.3.3 using the pre-installed public key of the DS that was called for the transaction.

Seq 5.94 **[Req 254]** The 3DS SDK shall generate JWE objects for the CReq messaging to the ACS utilising the keys derived from the 3DS SDK's own ephemeral key and the ACS Ephemeral key received from the 3DS Server.

Refer to Section 6.2.4 for detailed information about the JWS and JWE objects.

Upon receiving the CRes message from the ACS, the 3DS SDK displays the UI to the Cardholder for authentication and communicates the result back to the ACS in the CReq message.

Note: Several message interactions may occur as the Challenge is completed.

5.8 Browser-based Message Handling

5.8.1 3DS Method Handling

The 3DS Method allows for additional Browser information to be gathered by an ACS prior to receipt of the AReq message to help facilitate the transaction risk assessment. The use of the 3DS Method by an ACS is optional.

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

Seq 5.95 **[Req 255]** The 3DS Requestor and 3DS Server shall utilise the cached PRes message data to determine the ACS 3DS Method URL via the Card Range Data data element.

Note: Caching methods are implementation-specific and are outside of the scope of this *Core Specification*.

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

5.8.1.1 Recent Prior 3DS Method Call Does Not Exist

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

The 3DS Requestor shall:

Seq 5.96 **[Req 256]** For the card ranges that contain a 3DS Method URL in the cached PRes message, invoke the 3DS Method.

Seq 5.97 **[Req 257]** Invoke the 3DS Method call in advance of the AReq message for the same authentication transaction being sent.

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

Seq 5.98 **[Req 258]** Obtain from the 3DS Server or load from local cache, the 3DS Method URL if it exists for the card range.

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

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

Seq 5.99 **[Req 259]** The 3DS Method Data shall contain the following data elements (as specified in Table A.1 and Table A.2):

- 3DS Server Transaction ID (same as sent in the AReq message)
- 3DS Method Notification URL

Seq 5.100 **[Req 260]** Create a JSON object with the 3DS Method Data elements, then Base64url encode the JSON object.

Seq 5.101 **[Req 261]** Open a hidden HTML iframe in the Cardholder Browser with:

- the iframe attributes set as defined in Table A.23
- the sandbox attributes set as defined in Table A.24.

For Browser compatibility, iframe shall be made hidden with the following style setting: "visibility: hidden". For example: style="visibility:hidden".

Send a form with a field named `threeDSMethodData` containing the JSON Object via HTTP POST to the ACS 3DS Method URL obtained from the PRes message cache data.

The ACS shall:

Seq 5.102 **[Req 262]** Interact with the Cardholder Browser via the HTML iframe and then store the applicable values with the 3DS Server Transaction ID for use when the AReq message is received containing the same 3DS Server Transaction ID.

Seq 5.103 **[Req 263]** Recall the 3DS Server Transaction ID received in the initial 3DS Method POST, then Base64url encode the JSON object and send via a form with a field named `threeDSMethodData` in the Cardholder Browser HTML iframe using an HTTP POST method to the 3DS Method Notification URL. Refer to Table A.2 for detailed information about 3DS Method Data.

The 3DS Server shall:

Seq 5.104 **[Req 315]** Set the 3DS Method Completion Indicator = Y upon notification from the 3DS Requestor. If the 3DS Method does not complete within 5 seconds, set the 3DS Method Completion Indicator to = N.

Note: Upon notification that the 3DS Method has completed, the 3DS Server can submit the AReq message.

The ACS:

Seq 5.105 **[Req 264]** Should ensure that any action during the execution of the 3DS Method does not impact the user experience.

5.8.1.2 Recent Prior 3DS Method Call Does Exist

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

Seq 5.106 **[Req 415]** If the 3DS Requestor has already processed a 3DS Method call with the same Cardholder Account Number on the same device and Browser in the last 10 minutes, then the 3DS Requestor may use the previous 3DS method execution and choose not to invoke a new 3DS Method call.

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

5.8.2 Browser Challenge iframe Requirements

The Browser challenge will occur within the Cardholder Browser, and the ACS will provide a formatted challenge UI to the Cardholder within the Browser challenge iframe.

The 3DS Requestor shall:

- Seq 5.107 **[Req 265]** Select the size of the HTML iframe to be generated by the 3DS Requestor from one of the iframe sizes specified in the Challenge Window Size data element.
- Seq 5.108 **[Req 266]** Utilise a server authenticated TLS session as defined in Section 6.1.4.2.
- Seq 5.109 **[Req 267]** Create a 3-D Secure challenge iframe by generating a CReq message, creating an HTML iframe in the Cardholder Browser with the following settings:
- iframe attributes as defined in Table A.23.
 - sandbox attributes as defined in Table A.24.
- and generate an HTTP POST through the iframe to the ACS URL that was received in the ARes message.
- Seq 5.110 **[Req 268]** Post the CReq message containing the selected size in the Challenge Window Size data element to the ACS as defined in Table A.1.
- Seq 5.111 **[Req 324]** Provide a fallback mechanism for redirection in environments that do not support JavaScript.

Note: If the Cardholder initiates a page refresh, the 3DS Requestor repeats the previous steps starting from [Req 265] using the same iframe size and attributes.

The ACS shall:

- Seq 5.112 **[Req 269]** Receive the CReq message and respond with the HTML to render the challenge user interface within the iframe.

Note: During completion of the challenge by the Cardholder, there may be several interactions required.

After challenge completion, the ACS generates the RReq message. After receiving the corresponding RRes message, the ACS generates the CRes message and invokes the Browser to send an HTTP POST (for example, utilising JavaScript) to the Notification URL containing the CRes message as defined in Table A.1. This completes the challenge.

The 3DS Requestor shall:

- Seq 5.113 **[Req 270]** Close the challenge iframe upon receiving the CRes message by refreshing the parent page and removing the HTML iframe.

5.9 Message Error Handling

This section outlines the detailed error handling performed by a 3-D Secure component when an invalid message or Error Message is received.

The 3-D Secure component will receive and validate the message, verify and decrypt the message before performing further processing.

The validation process will check the message against the requirements for presence and format of each data element in the message as defined in Table A.1 and outlined in detail in Section 5.1.7.

The verification and decryption process will perform the cryptographic process against the message as defined in Chapter 6.

The outcome of the validation can be:

- Message is valid and therefore standard processing is followed, OR

- Message is invalid and therefore an exception scenario is processed, OR
- Message received is an Error Message and therefore an exception scenario is processed.

This section describes only the process where a message is invalid or an Error Message is received. The section follows the 3-D Secure transaction flow and provides a section for each combination of a 3-D Secure component receiving a specific message.

Valid message handling and exceptions based on business logic are described in Chapter 3.

5.9.1 DS AReq Message Error Handling

The DS processes the validation of the AReq message as follows:

- If message not recognised, the DS returns to the 3DS Server an Error Message (as defined in Section A.9) with Error Component = D and Error Code = 101.
- If an AReq message, the DS Validates the AReq message (as defined in Table B.1 and Section 5.1.7) according to the content of the Device Channel data element:
 - If any data element present fails validation, the DS returns to the 3DS Server an Error Message (as defined in Section A.9) with Error Component = D and Error Code = 203.
 - If any required data elements are missing, the DS returns to the 3DS Server an Error Message (as defined in Section A.9) with Error Component = D and Error Code = 201.
 - Otherwise, the message is not in error.

5.9.2 ACS AReq Message Error Handling

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

- If message not recognised, the ACS returns to the DS an Error Message (as defined in Section A.9) with Error Component = A and Error Code = 101.
- If AReq message, the ACS Validates the AReq message as defined in Table B.1 and Section 5.1.7 according to the content of the Device Channel data element:
 - If any data element present fails validation, the ACS returns to the DS an Error Message (as defined in Section A.9) with Error Component = A and Error Code = 203.
 - If any required data elements are missing, the ACS returns to the DS an Error Message (as defined in Section A.9) with Error Component = A and Error Code = 201.
 - Otherwise, the message is not in error.

5.9.3 DS ARes Message Error Handling

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

- For a message that cannot be recognised, the DS:
 - Returns to the ACS an Error Message (as defined in Section A.9) with Error Component = D and Error Code = 101.
 - Sends to the 3DS Server a message as defined in Section 5.9.3.1.

- For an ARes message, the DS Validates the ARes message as defined in Table B.2 and Section 5.1.7:
 - If any data element present fails validation, the DS:
 - Returns to the ACS an Error Message (as defined in Section A.9) with Error Component = D and Error Code = 203.
 - Sends to the 3DS Server a message as defined in Section 5.9.3.1.
 - If any required data elements are missing, the DS:
 - Returns to the ACS an Error Message (as defined in Section A.9) with Error Component = D and Error Code = 201.
 - Sends to the 3DS Server a message as defined in Section 5.9.3.1.
 - Otherwise, the message is not in error.
- For an Error Message, the DS sends to the 3DS Server a message as defined in Section 5.9.3.2.

To finalise, the DS logs transaction information as required by DS rules.

5.9.3.1 Message in Error

The DS:

If a message is in error and a specific transaction can be identified, the DS sends to the 3DS Server using the secure link established in **[Req 12]** for an app-based transaction, **[Req 90]** for a Browser-based transaction, or **[Req 275]** for a 3RI transaction, EITHER:

- an Error Message (as defined in Section A.9) with Error Component = D and the Error Code returned to ACS, OR
- an ARes message (as defined in Table B.2) with Transaction Status set to the appropriate value as defined by the specific DS.

5.9.3.2 Error Message Received

The DS:

If an error message is received and a specific transaction can be identified, the DS sends to the 3DS Server using the secure link established in **[Req 12]** for an app-based transaction, **[Req 90]** for a Browser-based transaction, or **[Req 275]** for a 3RI transaction, EITHER:

- an Error Message as received from the ACS, OR
- an ARes message (as defined in Table B.2) with Transaction Status set to the appropriate value as defined by the specific DS.

5.9.4 3DS Server ARes Message Error Handling

The 3DS Server processes the validation of the ARes message or Error Message as follows:

- For a message that cannot be recognised, the 3DS Server:
 - Returns to the DS an Error Message (as defined in Section A.9) with Error Component = S and Error Code = 101.
 - If a specific transaction can be identified, informs the 3DS Requestor Environment that an error occurred.
- For an ARes message, the 3DS Server Validates the ARes message as defined in Table B.2 and Section 5.1.7:
 - If any data element present fails validation, the 3DS Server:

- Returns to the DS an Error Message (as defined in Section A.9) with Error Component = S and Error Code = 203.
- If a specific transaction can be identified, informs the 3DS Requestor Environment that an error occurred.
- If any required data elements are missing, the 3DS Server:
 - Returns to the DS an Error Message (as defined in Section A.9) with Error Component = S and Error Code = 201.
 - If a specific transaction can be identified, informs the 3DS Requestor Environment that an error occurred.
- Otherwise, the message is not in error.
- For an Error Message, if a specific transaction can be identified, the 3DS Server informs the 3DS Requestor Environment that an error occurred.

Note: For App-based implementations, the 3DS Requestor Environment is expected to inform the 3DS SDK about the error.

5.9.5 ACS CReq Message Error Handling—01-APP

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

- For a message, which the ACS cannot verify or decrypt correctly as defined in Section 6.2.4.3 or which the ACS cannot recognise, the ACS:
 - Returns to the 3DS SDK an Error Message (as defined in Section A.9) with Error Component = A and Error Code = 101 (not recognised) or 302 (verification or decryption failure) using the secure link established in **[Req 44]**.
 - If a specific transaction can be identified, sends to the DS an RReq message as defined in Section 5.9.5.1.
- For a correctly verified, decrypted, and recognised CReq message, the ACS Validates the CReq message as defined in Table B.3 and Section 5.1.7 according to the Device Channel = 01-APP:
 - If any data element present fails validation, the ACS:
 - Returns to the 3DS SDK an Error Message (as defined in Section A.9) with Error Component = A and Error Code = 203 using the secure link established in **[Req 44]**.
 - If a specific transaction can be identified, sends to the DS an RReq message as defined in Section 5.9.5.1.
 - If any required data elements are missing, the ACS:
 - Returns to the 3DS SDK an Error Message (as defined in Section A.9) with Error Component = A and Error Code = 201 using the secure link established in **[Req 44]**.
 - If a specific transaction can be identified, sends to the DS an RReq message as defined in Section 5.9.5.1.
 - If SDK Type = 02 (as received in the AReq message) AND the CReq message is not protected using A128GCM ("enc" as defined in Section 6.2.4.3 is not A128GCM), the ACS:

- Returns to the 3DS SDK an Error Message (as defined in Section A.9) with Error Component = A and Error Code = 310 using the secure link established in [Req 44].
- If a specific transaction can be identified, sends to the DS an RReq message as defined in Section 5.9.5.1.
 - Otherwise, the message is not in error.
- For an Error Message, if a specific transaction can be identified, the ACS:
 - Establishes a secure link with the DS as defined in Section 6.1.3.2.
 - Sends to the DS an RReq message (as defined in Table B.6) with Transaction Status = U and Challenge Cancelation Indicator = 09 using the secure link.

5.9.5.1 Message in Error

The ACS:

- Establishes a secure link with the DS as defined in Section 6.1.3.2.
- Sends to the DS an RReq message (as defined in Table B.8) with Transaction Status = U and Challenge Cancelation Indicator = 10 using the secure link.

5.9.6 ACS CReq Message Error Handling—02-BRW

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

- For a message that cannot be recognised, the ACS:
 - If a specific transaction can be identified, sends to the DS an RReq message (as defined in Section 5.9.5.1).
- For multiple received CReq messages, the ACS:
 - Validates that the data elements from the CReq message are identical to the first CReq message.
 - If any data element present is different, the ACS:
 - Sends to the DS an RReq message (as defined in Section 5.9.5.1).
 - Returns (via the Browser) an Error Message (as defined in Section A.9) with Error Component = A and Error Code = 305.
 - If the subsequent CReq message is identical to the first CReq message, but the ACS does not proceed with the challenge, the ACS:
 - Sends to the DS an RReq message (as defined in Section 5.9.5.1).
 - Returns an Error message to the Notification URL (via the Browser) with Error Component = A and Error Code = 314.
 - If the subsequent CReq message is received after the ACS has sent the RReq message, the ACS:
 - Returns an Error message to the Notification URL (via the Browser) with Error Component = A and Error Code = 315.
- For a CReq message, the ACS validates the CReq message (as defined in Table B.3 and Section 5.1.7) according to the Device Channel = 02-BRW:
 - If any data element present fails validation, the ACS:

- If the Notification URL is valid, returns (via the Browser) an Error Message (as defined in Section A.9) with Error Component = A and Error Code = 203.
- If a specific transaction can be identified, sends to the DS an RReq message (as defined in Section 5.9.5.1).
- If any required data elements are missing, the ACS:
 - If the Notification URL is present, returns (via the Browser) an Error Message (as defined in Section A.9) with Error Component = A and Error Code = 201.
 - If a specific transaction can be identified, sends to the DS an RReq message (as defined in Section 5.9.5.1).
- Otherwise, the message is not in error.

5.9.7 3DS SDK CRes Message Error Handling

The 3DS SDK processes the validation of the CRes message or Error Message as follows:

- For a message that the 3DS SDK cannot verify or decrypt correctly as defined in Section 6.2.4.2, or cannot recognise, the 3DS SDK returns to the ACS an Error Message (as defined in Section A.9) with Error Component = C and Error Code = 101 (not recognised) or 302 (verification or decryption failure).
- For a correctly verified, decrypted and recognised CRes message, the 3DS SDK Validates the CRes message (as defined in Table B.4 and Section 5.1.7):
 - If any data element present fails validation, the 3DS SDK returns to the ACS an Error Message (as defined in Section A.9) with Error Component = C and Error Code = 203.
 - If any required data elements are missing, the 3DS SDK returns to the ACS an Error Message (as defined in Section A.9) with Error Component = C and Error Code = 201.
 - Otherwise, the message is not in error.
- For an Error Message, the 3DS SDK informs the 3DS Requestor Environment that an error occurred.

5.9.8 DS RReq Message Error Handling

The DS processes the validation of the RReq message as follows:

- For a message that cannot be recognised, the DS:
 - Returns to the ACS an Error Message (as defined in Section A.9) with Error Component = D and Error Code = 101.
 - If a specific transaction can be identified, sends to the 3DS Server an Error Message (as defined in Section 5.9.8.1).
- For an RReq message, the DS Validates the RReq message (as defined in Table B.8 and Section 5.1.7):
 - If any data element present fails validation, the DS:
 - Returns to the ACS an Error Message (as defined in Section A.9) with Error Component = D and Error Code = 203.

- If a specific transaction can be identified, sends to the 3DS Server an Error Message (as defined in Section 5.9.8.1).
- If any required data elements are missing, the DS:
 - Returns to the ACS an Error Message (as defined in Section A.9) with Error Component = D and Error Code = 201.
 - If a specific transaction can be identified, sends to the 3DS Server an Error Message (as defined in Section 5.9.8.1).
- Otherwise, the message is not in error.

5.9.8.1 Message in Error

The DS:

- Establishes a secure link with the 3DS Server (as defined in Section 6.1.2.2) using the 3DS Server URL extracted from the AReq message and stored in:
 - **[Req 22]** for an App-based transaction OR
 - **[Req 99]** for a Browser-based transaction OR
 - **[Req 427]** for a 3RI-based transaction
- Sends to the 3DS Server an Error Message (as defined in Section A.9) with Error Component = D and the same Error Code that was returned to the ACS.

5.9.9 3DS Server RReq Message Error Handling

The 3DS Server processes the validation of the RReq message or Error Message as follows:

- For a message that cannot be recognised, the 3DS Server:
 - Returns to the DS an Error Message (as defined in Section A.9) with Error Component = S and Error Code = 101.
 - If a specific transaction can be identified, informs the 3DS Requestor Environment that an error occurred.
- For an RReq message, the 3DS Server Validates the RReq message (as defined in Table B.8 and Section 5.1.7):
 - If any data element present fails validation, the 3DS Server:
 - Returns to the DS an Error Message (as defined in Section A.9) with Error Component = S and Error Code = 203.
 - If a specific transaction can be identified, informs the 3DS Requestor Environment that an error occurred.
 - If any required data elements are missing, the 3DS Server:
 - Returns to the DS an Error Message (as defined in Section A.9) with Error Component = S and Error Code = 201.
 - If a specific transaction can be identified, informs the 3DS Requestor Environment that an error occurred.
 - Otherwise, the message is not in error.
- For an Error Message, if a specific transaction can be identified, the 3DS Server informs the 3DS Requestor Environment that an error occurred.

5.9.10 DS RRes Message Error Handling

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

- For a message that cannot be recognised, the DS:
 - Returns to the 3DS Server an Error Message (as defined in Section A.9) with Error Component = D and Error Code = 101.
 - If a specific transaction can be identified, sends to the ACS an Error Message (as defined in Section A.9) with Error Component = D and Error Code = 101 using the secure link established in **[Req 63]** for an app-based transaction or **[Req 125]** for a Browser-based transaction, or **[Req 350]** for a 3RI transaction.
- For an RRes message, the DS Validates the RRes message (as defined in Table B.9 and Section 5.1.7):
 - If any data element present fails validation, the DS:
 - Returns to the 3DS Server an Error message (as defined in Section A.9) with Error Component = D and Error Code = 203.
 - If a specific transaction can be identified, sends to the ACS an Error Message (as defined in Section A.9) with Error Component = D and Error Code = 203 using the secure link established in **[Req 63]** for an app-based transaction or **[Req 125]** for a Browser-based transaction, or **[Req 350]** for a 3RI transaction.
 - If any required data elements are missing, the DS:
 - Returns to the 3DS Server an Error message (as defined in Section A.9) with Error Component = D and Error Code = 201.
 - If a specific transaction can be identified, sends to the ACS an Error Message (as defined in Section A.9) with Error Component = D and Error Code = 201 using the secure link established in **[Req 63]** for an app-based transaction or **[Req 125]** for a Browser-based transaction, or **[Req 350]** for a 3RI transaction.
 - Otherwise, the message is not in error.
- For an Error message, if a specific transaction can be identified, the DS sends to the ACS the Error Message as received from the 3DS Server using the secure link established in **[Req 63]** for an app-based transaction or **[Req 125]** for a Browser-based transaction, or **[Req 350]** for a 3RI transaction.

To finalise, the DS logs transaction information as required by DS rules.

5.9.11 ACS RRes Message Error Handling—01-APP

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

- For a message that cannot be recognised, the ACS:
 - Returns to the DS an Error Message (as defined in Section A.9) with Error Component = A and Error Code = 101.
 - If a specific transaction can be identified and the transaction is not a Decoupled Authentication transaction, the ACS:

- Sends to the 3DS SDK an Error Message (as defined in Section A.9) with Error Component = A and Error Code = 101 using the secure link established in **[Req 56]**.
- For an RRes message, the ACS Validates the RRes message (as defined in Table B.9 and Section 5.1.7):
 - If any data element present fails validation, the ACS:
 - Returns to the DS an Error message (as defined in Section A.9) with Error Component = A and Error Code = 203.
 - If a specific transaction can be identified and the transaction is not a Decoupled Authentication transaction, sends to the 3DS SDK an Error Message (as defined in Section A.9) with Error Component = A and Error Code = 203 using the secure link established in **[Req 56]**.
 - If any required data elements are missing, the ACS:
 - Returns to the DS an Error message (as defined in Section A.9) with Error Component = A and Error Code = 201.
 - If a specific transaction can be identified and the transaction is not a Decoupled Authentication transaction, sends to the 3DS SDK an Error Message (as defined in Section A.9) with Error Component = A and Error Code = 201 using the secure link established in **[Req 56]**.
 - Otherwise, the RRes message is not in error. Note: Transaction Status in the CRes message was then determined by the ACS in Section 3.1, Step 17.
- For an Error message, if a specific transaction can be identified and the transaction is not a Decoupled Authentication transaction, the ACS sends to the 3DS SDK an Error Message (as defined in Section A.9) with Error Component = A and Error Code = 403 using the secure link established in **[Req 56]**.

5.9.12 ACS RRes Message Error Handling—02-BRW

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

- For a message that cannot be recognised, the ACS:
 - Returns to the DS an Error Message (as defined in Section A.9) with Error Component = A and Error Code = 101.
 - If a specific transaction can be identified and the transaction is not a Decoupled Authentication transaction, the ACS sends to the 3DS Server (via Browser) a CRes message.
- For an RRes message, the ACS Validates the RRes message (as defined in Table B.9 and Section 5.1.7):
 - If any data element present fails validation, the ACS:
 - Returns to the DS an Error message (as defined in Section A.9) with Error Component = A and Error Code = 203.
 - If a specific transaction can be identified and the transaction is not a Decoupled Authentication transaction, sends to the 3DS Server (via Browser) a CRes message.
 - If any required data elements are missing, the ACS:

- Returns to the DS an Error message (as defined in Section A.9) with Error Component = A and Error Code = 201.
- If a specific transaction can be identified and the transaction is not a Decoupled Authentication transaction, sends to the 3DS Server (via Browser) a CRes message.
 - Otherwise, the message is not in error. Note: Transaction Status in the CRes message was then determined by the ACS in Section 3.3, Step 15.
- For an Error message, if a specific transaction can be identified and the transaction is not a Decoupled Authentication transaction, the ACS sends to the 3DS Server (via Browser) a CRes message.

5.9.13 ACS RRes Message Error Handling—03-3RI

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

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

5.10 UTC Date and Time

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

Seq 5.114 **[Req 416]** The maximum desynchronisation with UTC time for the 3DS Server, DS and ACS shall be less than 15 minutes when providing or verifying the UTC date and time data elements.

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

Seq 5.115 **[Req 417]** The ACS may respond to the DS with an Error Message with Error Component = A and Error Code = 203 if the SDK Signature Timestamp or Purchase Date & Time data elements are received with a UTC time difference greater than 30 minutes.

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

5.11 OReq/ORes Message Handling Requirements

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

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

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

The DS using Operation Messages shall:

Seq 5.116 **[Req 435]** Prepare and send the OReq message or the sequence of OReq messages in the sequence number order via a secure link with the recipient (3DS Server or ACS) established as defined in Table B.10.

Seq 5.117 **[Req 436]** Immediately retry a connection upon any failure to complete the initial TCP/IP connection and TLS handshake to the recipient.

Seq 5.118 **[Req 437]** Upon the second failure to complete the TCP/IP connection and TLS handshake to the recipient, end the transaction, and periodically retry within the 24-hour window at 60-second intervals until the transaction completes successfully.

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

Seq 5.119 **[Req 438]** Receive and validate all the OReq messages in the sequence as defined in Table B.10:

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

Seq 5.120 **[Req 439]** Prepare and send the ORes message to the DS via the secure link as defined in Table B.11.

The DS using Operation Messages shall:

Seq 5.121 **[Req 440]** Receive and validate the ORes message as defined in Table B.11:

- If any data element present fails validation, the DS:

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

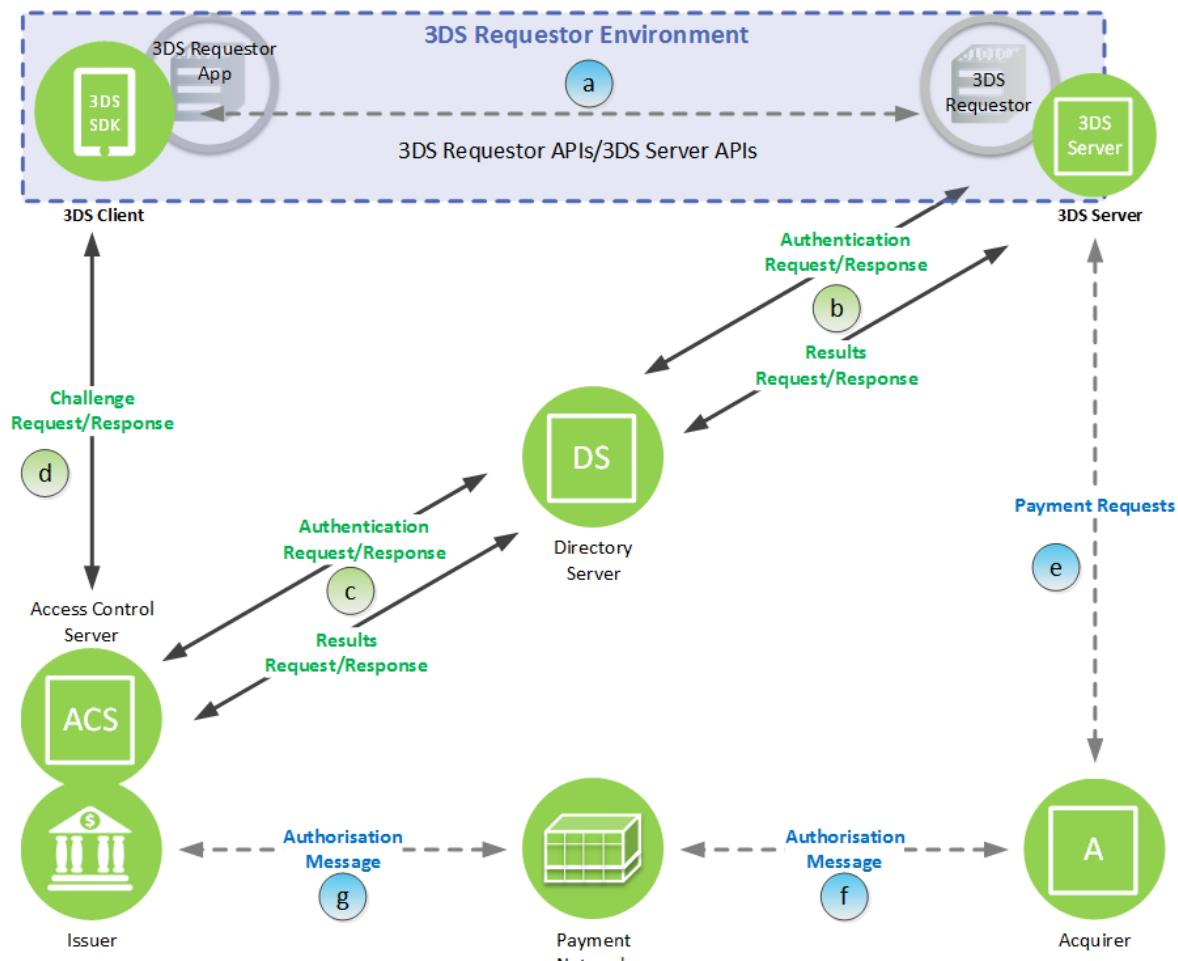
Note: If there is more than one OReq message in the message sequence, the DS and the OReq recipient repeat the previous steps starting from [Req 435].

Note: 3-D Secure processing completes.

6 EMV 3-D Secure Security Requirements

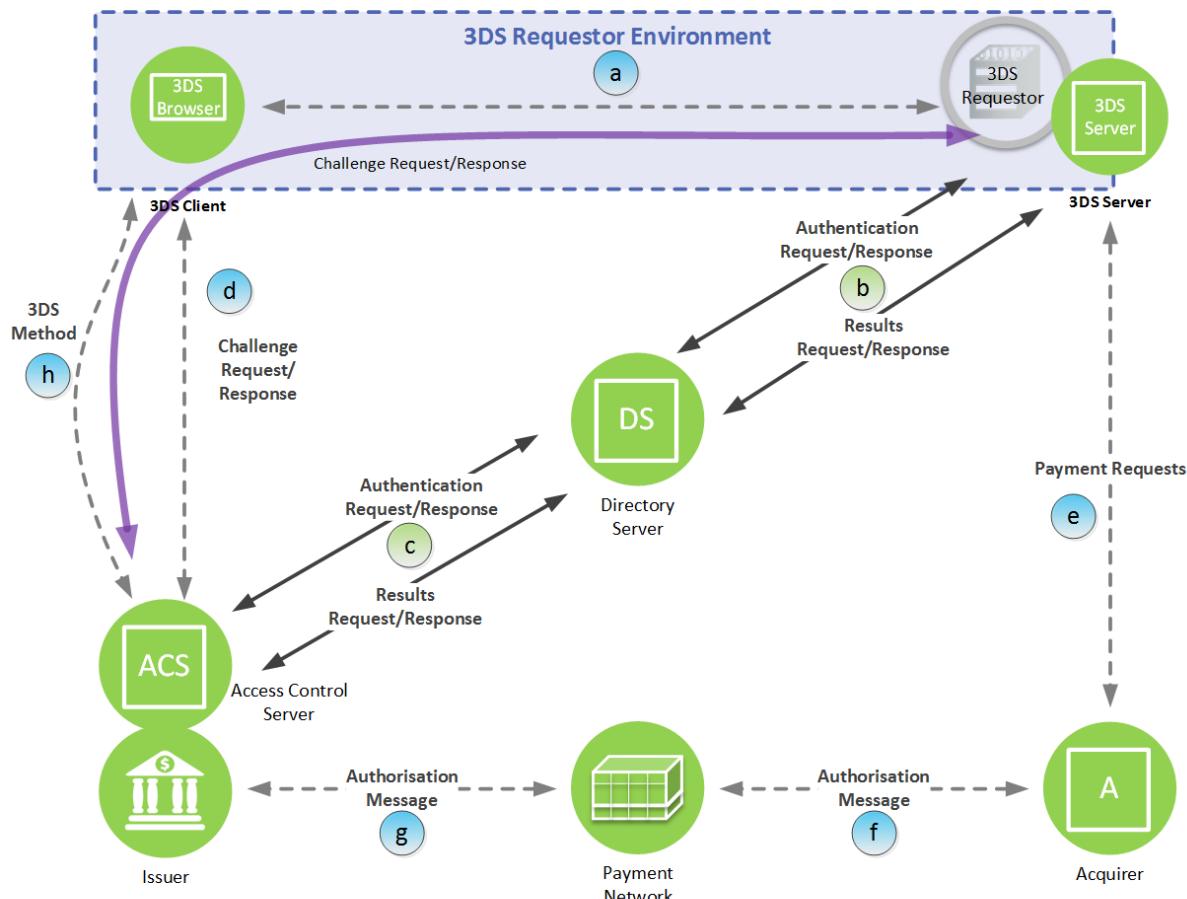
This chapter provides security detail for the EMV 3-D Secure security requirements referenced within this specification. Specifically, it describes the expected features of the links between the 3-D Secure components and the 3-D Secure security functions. The characters in both Figure 6.1 and Figure 6.2 refer to the specific link as defined in the following sections.

Figure 6.1: Security Flow—App-based



Note: Dashed arrows and 3DS Requestor are not part of 3DS specification and are shown for clarity only

Figure 6.2: Security Flow—Browser-based



Note: Dashed arrows and 3DS Requestor are not part of 3DS specification and are shown for clarity only

6.1 Links

For each link, the following expectations apply:

6.1.1 Link a: Consumer Device—3DS Requestor

The Consumer Device to 3DS Requestor link is within the 3DS Requestor Environment between the 3DS Requestor App and the 3DS Requestor. The 3DS Requestor link established between the 3DS Requestor and the Consumer device uses a TLS Internet protocol as part of the interaction between the 3DS Requestor App or the Browser on the Consumer Device and the 3DS Requestor. Further links may be required between the Browser and the 3DS Requestor Environment if the URLs for return of the 3DS Method Notification or the CRes are different than the initiator endpoints.

When the Cardholder interaction with the 3DS Requestor moves to 3-D Secure protocol-specific actions, the links will need to be in a secured state. This will be 3DS Requestor-specific, with the expectation that it satisfies Payment System security requirements with at least a TLS protocol with 3DS Requestor (server) authentication by the 3DS Requestor App or the Browser.

If the 3DS Requestor and 3DS Server are separate components, data transferred between the components needs to be protected at a level that satisfies Payment System security requirements with mutual authentication of both servers.

In some implementations, the customer through the 3DS Requestor App or the Browser will have logged on to the 3DS Requestor to start interaction, and the link may have met the 3DS Requestor's security requirements from the start. The level of customer authentication delivered by such a process is carried in the 3DS Requestor Authentication Information data element.

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

6.1.2 Link b: 3DS Server—DS

6.1.2.1 For AReq/ARes

The 3DS Server to DS link for the AReq/ARes messages is established using a TLS protocol with mutual authentication. The public key certificates of both parties are signed by the DS CA, with the 3DS Server making the necessary selection if it connects to more than one DS.

- Protocol—TLS using cipher suite as described in Annex D
- 3DS Server public key Pb_R
 - Client Certificate format: X.509
- DS public key: Pb_{DS}
 - Server Certificate format: X.509
- CA signing 3DS Server key—DS CA
- CA signing DS key—DS CA

6.1.2.2 For RReq/RRes

The DS to 3DS Server link for the RReq/RRes messages is established using a TLS protocol with mutual authentication. The public key certificates of both parties are signed by the DS CA.

- Protocol—TLS using cipher suite as described in Annex D
- DS public key: Pb_{DS}
 - Client Certificate format: X.509
- 3DS Server public key Pb_R
 - Server Certificate format: X.509
- CA signing DS key—DS CA
- CA signing 3DS Server key—DS CA

6.1.3 Link c: DS—ACS

6.1.3.1 For AReq/ARes

The DS to ACS link for the AReq/ARes messages is established using a TLS protocol with mutual authentication. The public key certificates of both parties are signed by the DS CA.

- Protocol—TLS using cipher suite as described in Annex D
- DS public key Pb_{DS}
 - Client Certificate format: X.509

- ACS public key Pb_{ACS}
 - Server Certificate format: X.509
- CA signing DS key—DS CA
- CA signing ACS key—DS CA

6.1.3.2 For RReq/RRes

The ACS to DS link for the RReq/RRes messages is established using a TLS protocol with mutual authentication. The public key certificates of both parties are signed by the DS CA.

- Protocol—TLS using cipher suite as described in Annex D
- ACS public key Pb_{ACS}
 - Client Certificate format: X.509
- DS public key Pb_{DS}
 - Server Certificate format: X.509
- CA signing ACS key—DS CA
- CA signing DS key—DS CA

6.1.4 Link d: 3DS Client—ACS

6.1.4.1 For App-based CReq/CRes

For the App-based protocol, the direct link between the 3DS SDK and the ACS is only established if the transaction requires a challenge. It is initiated by the 3DS SDK using the URL provided to it in the ARes and established using a TLS protocol with ACS (server) authentication by the 3DS SDK. The public key certificate for the ACS is signed by a commercial CA.

- Protocol—TLS 1.2 or higher
- ACS public key—commercial
 - Certificate format: commercial
- CA signing ACS key—commercial CA

The challenge and Cardholder response data is encrypted and MACed using the session keys previously established between the ACS and the 3DS SDK.

- Protocol—secure channel as described in Section 6.2.4 using the previously established session keys.

If the CRes message contains a URL(s) directing the 3DS SDK to fetch data from an external server, an additional link is established using a TLS protocol, with server authentication by the 3DS SDK based on a commercial server certificate.

6.1.4.2 For Browser-based CReq/CRes

For the Browser-based protocol, the direct link between the Browser and the ACS is only established if the transaction requires a challenge. It is initiated by the Browser from an iframe using the URL provided to it in the ARes and established using a TLS protocol with ACS (server) authentication by the Browser. The public key certificate for the ACS is signed by a commercial CA.

- Protocol—TLS 1.2 or higher
- ACS public key—commercial

- Certificate format: commercial
- CA signing ACS key—commercial CA

6.1.5 Link e: 3DS Integrator—Acquirer (Payment Authentication only)

The 3DS Integrator to Acquirer link is the regular payment link for the Merchant. No additional security requirements are set by 3-D Secure.

6.1.6 Link f: Acquirer—Payment System (Payment Authentication only)

The Acquirer to Payment System link is across the regular Payment System network for the Payment System involved. No additional security requirements are set by 3-D Secure.

6.1.7 Link g: Payment System—Issuer (Payment Authentication only)

The Payment System to Issuer link is across the regular Payment System network for the Payment System involved. No additional security requirements are set by 3-D Secure.

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

The link between the Browser and the ACS for the 3DS Method is opened from a hidden iframe loaded by the 3DS Server as part of the checkout page. It is used for the ACS to gather Device Information to be returned to the ACS. This includes the 3DS Server Transaction ID which enables the ACS to marry the information to the correct transaction. Note that the URL of the ACS used for the 3DS Method is regarded as different from the URL of the ACS for the Challenge Flow and separate TLS links will be established for the 3DS Method and for any Challenge Flow.

The link is established using a TLS protocol with server authentication of the ACS by the Browser. The public key certificate for the ACS is signed by a commercial CA.

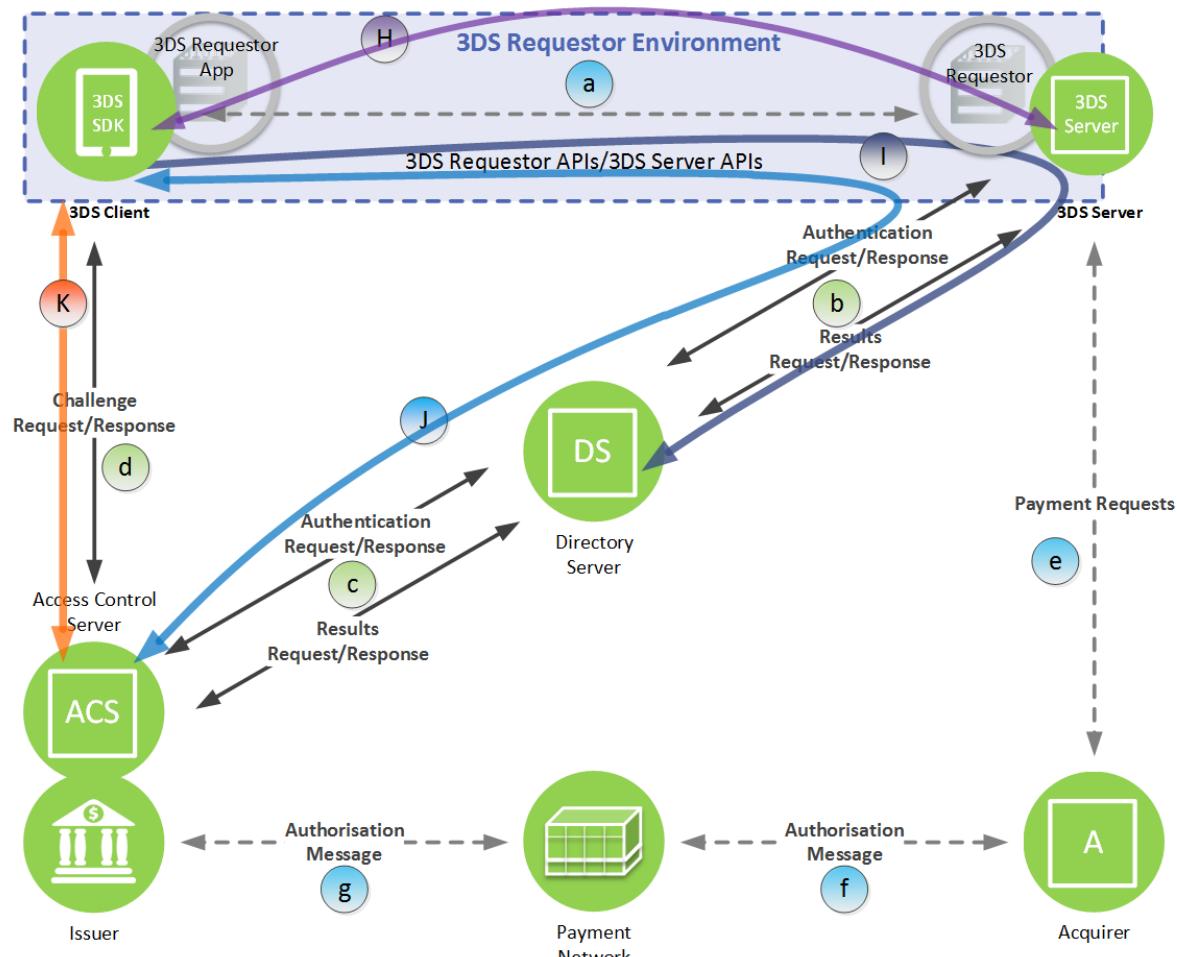
- Protocol—TLS 1.2 or higher
- ACS public key—commercial
 - Certificate format: commercial
- CA signing ACS key—commercial CA

Note: For 3RI transactions, only the 3DS Server, DS and ACS with links b and c for AReq/ARes apply.

6.2 Security Functions

In addition to the internet-based links there are a number of security functions that are specific to the App-based flow.

Figure 6.3: Security Functions



Note: Dashed arrows and 3DS Requestor are not part of 3DS specification and are shown for clarity only

6.2.1 Function H: Authenticity of the 3DS SDK

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

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

These 3DS SDK to DS functions occur via the 3DS Server. The purpose is to allow:

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

6.2.2.1 3DS SDK Device Information Encryption

As a prerequisite, the 3DS SDK is loaded with (or has access to) the public key P_{DS} for the DS. There may be multiple keys for each DS with the keys each having an individual identifier PDSid. A UUID format is recommended for the PDSid.

The 3DS SDK:

- Identifies the DS public key P_{DS} , associated attributes, and encryption function (relating to the BIN and optionally other information) from information provided by the 3DS Requestor Environment. If the public key cannot be identified, ceases processing and report error.
- Creates a JSON Object of Device Information.
- If P_{DS} is an RSA key:
 - Encrypts the JSON object according to JWE (RFC 7516) using JWE Compact Serialization. The parameter values for this version of the specification and to be included in the JWE protected header are:
 - "alg": "RSA-OAEP-256"
 - "kid":PDSid
 - "enc": "A128CBC-HS256 or A128GCM"
 - All other parameters optional
- Else if P_{DS} is an ECC key:
 - Encrypts the JSON object as follows:
 - Generates a fresh ephemeral key pair (Q_{SDK}, d_{SDK}) as described in Annex C.
 - Conducts a DH key exchange process according to JWA (RFC 7518) in Direct Key Agreement mode using ECDH-ES, curve P-256, d_{SDK} and P_{DS} with Concat KDF to produce a 256-bit CEK. The Concat KDF parameter values for this version of the specification are:
 - Keydatalen = 256
 - AlgorithmID = empty string (length = 0x00000000)
 - PartyUIInfo = empty string (length = 0x00000000)
 - PartyVIInfo = directoryServerID (length || ascii string)
 - SuppPublInfo = Keydatalen (0x00000100)
 - SuppPrivInfo = empty octet sequence
 - Generates 128-bit random data as IV (included in the JWE)
 - Encrypts the JSON object according to JWE (RFC 7516) using the CEK and JWE Compact Serialization. The parameter values for this version of the specification and to be included in the JWE protected header are:
 - "alg": "ECDH-ES"

- "kid":PDSid
- "epk": Q_{SDK} ,
{"kty": "EC",
"crv": "P-256"
"x":x coordinate of Q_{SDK}
"y":y coordinate of Q_{SDK} }
- "enc":either "A128CBC-HS256" or "A128GCM"
- All other parameters: optional
- If the algorithm is A128CBC-HS256 use the full CEK or if the algorithm is A128GCM use the leftmost 128 bits of the CEK.
- Deletes the ephemeral key pair (Q_{SDK} , d_{SDK})
- Makes the resulting JWE available to the 3DS Server as SDK Encrypted Data.

6.2.2.2 DS Device Information Decryption

The DS:

- If the protected header of the JWE in the SDK Encrypted Data field indicates that RSA-OAEP-256 was used for encryption:
 - Decrypts the SDK Encrypted Data field from the AReq message according to JWE (RFC 7516) using the parameter values from the protected header (RSA-OAEP-256 and P_{DS} as indicated by "kid") either A128CBC-HS256 or A128GCM as indicated by the "enc" parameter in the protected header.
- Else, if the protected header of the JWE in the SDK Encrypted Data field indicates that ECDH-ES was used for encryption:
 - Conducts a DH key exchange process according to JWA (RFC 7518) in Direct Key Agreement mode using the parameter values from the protected header (ECDH-ES, curve P-256, Q_{SDK} , and d_{DS} by "kid") and Concat KDF to produce a 256-bit CEK. The Concat KDF parameter values for this version of the specification are:
 - Keydatalen = 256
 - AlgorithmID = empty string (length = 0x00000000)
 - PartyUInfo = empty string (length = 0x00000000)
 - PartyVInfo = directoryServerID (length || ascii string)
 - SuppPubInfo = Keydatalen (0x00000100)
 - SuppPrivInfo = empty octet sequence
 - Decrypt the JWE in the SDK Encrypted Data field according to JWE (RFC 7516) using the CEK and either A128CBC-HS256 or A128GCM as indicated by the "enc" parameter in the protected header. If the algorithm is A128GCM the leftmost 128bits of CEK is used. If decryption fails, ceases processing and reports error.
- Insert the result into Device Information for the AReq message to the ACS.

6.2.2.3 Split-SDK Server Signature

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

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

The Split-SDK Server:

- Creates a JSON object of the following data as the JWS payload to be signed:
 - SDK Reference Number
 - SDK Transaction ID
 - Split-SDK Server ID
 - SDK Signature Timestamp
- Generates a digital signature of the full JSON object according to JWS (RFC 7515) using JWS Compact Serialization. The parameter values for this version of the specification and to be included in the JWS header are:
 - "alg": PS256⁷ or ES256
 - "x5c": X.5C v3: Cert (Pb_{SDK}) and chaining certificates, if present (without any DS CA public key certificate)
- All other parameters: optional
- Includes the resulting JWS in the AReq message as SDK Server Signed Content

6.2.2.4 DS Verification of Split-SDK Server Signed Content

The DS:

Using the CA public key of the DS CA, validates the JWS from the Split-SDK Server according to JWS (RFC7515) using either PS256 or ES256 as indicated by the "alg" parameter in the header. If validation fails, ceases processing and reports error. If the Split-SDK Server Signature is valid, the DS has confirmed the authenticity of the Split-SDK Server, that the SDK reference number and date are correct.

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

Using data transferred in the AReq/ARes messages the 3DS SDK and ACS execute a DH key exchange protocol to establish keys for a secure channel that will later be used to protect the CReq/CRes messages in the Challenge Flow if the transaction is challenged.

6.2.3.1 3DS SDK Preparation for Secure Channel

The 3DS SDK:

- Generates a fresh ephemeral key pair (Q_C, d_C) as described in Annex C and provides Q_C as a JWK for inclusion in the AReq message as `sdkEphemPubKey`.

6.2.3.2 ACS Secure Channel Setup

If the ACS determines that a challenge is required to secure the direct link between the 3DS SDK and ACS for the CReq/CRes messages, it completes the security function during the AReq/ARes exchange as follows:

As a prerequisite, the ACS has a key pair (Pb_{ACS}, Pv_{ACS}) with a certificate Cert (Pb_{ACS}). This certificate is an X.509 certificate signed by a DS CA whose public key is known to the 3DS SDK.

⁷ PS256 (RSA-PSS) is specified in preference to RS256 (RSASSA-PKCS1-v1_5) following the recommendation in RFC 3447 (2003).

The ACS receives Q_C from the 3DS SDK in the AReq message (via the 3DS Server and the DS).

The ACS signs its own ephemeral public key Q_T together with Q_C received from the 3DS SDK and the ACS URL (to be used by the 3DS SDK for the CReq message). The resulting signature is sent back to the 3DS SDK together with Cert (Pb_{ACS}) for the ACS public key.

The ACS:

- Generates a fresh ephemeral key pair (Q_T, d_T) as described in Annex C.
- Checks that Q_C is a point on the curve P-256.
- Completes the DH key exchange process as a local mechanism according to JWA (RFC 7518) in Direct Key Agreement mode using ECDH-ES, curve P-256, d_T and Q_C with Concat KDF to produce a 256-bit CEK which is identified to the ACS Transaction ID. The Concat KDF parameter values for this version of the specification are:
 - Keydatalen = 256
 - AlgorithmID = empty string (length = 0x00000000)
 - PartyUInfo = empty string (length = 0x00000000)
 - PartyVInfo = sdkReferenceNumber (length || ascii string)
 - SuppPubInfo = Keydatalen (0x00000100)
 - SuppPrivInfo = empty octet sequence
 - CEK: 256 bits allocated as:
 - CEK_{A-S} : 256 bits
 - CEK_{S-A} : 256 bits

Note: Key separation is good practice for opposite directions of the 3DS SDK to ACS link. In this version of the specification CEK_{A-S} and CEK_{S-A} are extracted with the same value. Thus, for A128CBC-HS256 the same 256-bit key will be used in both directions. For A128GCM the key is split as two 128-bit components, resulting in separate keys for each direction.

- Creates a JSON object of the following data as the JWS payload to be signed:
 - {"acsEphemPubKey": Q_T , "sdkEphemPubKey": Q_C , "acsURL": "https://mybank.com/acs"}
- Generates a digital signature of the full JSON object according to JWS (RFC 7515) using JWS Compact Serialization. The parameter values for this version of the specification and to be included in the JWS header are:
 - "alg": PS256⁸ or ES256
 - "x5c": X.5C v3: Cert(Pb_{ACS}) and chaining certificates, if present (without any DS CA public key certificate)
 - All other parameters: optional
- Includes the resulting JWS in the ARes message as ACS Signed Content

⁸ PS256 (RSA-PSS) is specified in preference to RS256 (RSASSA-PKCS1-v1_5) following the recommendation in RFC 3447 (2003).

- Deletes the ephemeral key pair (Q_T, d_T)
- Zeros the channel counters ACSCounterAtoS (8 bits) and ACSCounterStoA (8 bits)

6.2.3.3 3DS SDK Secure Channel Setup

The 3DS SDK receives the necessary data elements from the ARes message (extracted by the 3DS Server), including Q_T , ACS Signature, ACS Public Key Certificate, and ACS URL. The 3DS SDK:

- Using the CA public key of the DS CA identified from information provided by the 3DS Server, validates the JWS from the ACS according to JWS (RFC7515) using either PS256 or ES256 as indicated by the "alg" parameter in the header. If validation fails, ceases processing and report error.
- Verifies that the SDK Qc present in the signature is the same as generated by the SDK in Section 6.2.3.1, and provided for inclusion in the AReq message as sdkEphemPubKey.
- Completes the DH key exchange process according to JWA (RFC 7518) in Direct Key Agreement mode using curve P-256, d_C and Q_T , with Concat KDF to produce a 256-bit CEK, which is identified to the ACS Transaction ID received in the ARes message. The Concat KDF parameter values for this version of the specification are:
 - Keydatalen = 256
 - AlgorithmID = empty string (length = 0x00000000)
 - PartyUInfo = empty string (length = 0x00000000)
 - PartyVInfo = sdkReferenceNumber (length || ascii string)
 - SuppPubInfo = Keydatalen (0x00000100)
 - SuppPrivInfo = empty octet sequence
 - CEK: 256 bits allocated as:
 - CEK_{A-S}: 256 bits
 - CEK_{S-A}: 256 bits
- Deletes the ephemeral key pair (Q_C, d_C)
- Zeros the channel counters SDKCounterAtoS (8 bits) and SDKCounterStoA (8 bits)

If the ACS signature is valid, the 3DS SDK has confirmed the authenticity of the ACS, that the session keys are fresh, and that the ACS URL is correct.

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

6.2.4.1 3DS SDK—CReq

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

- Creates a JSON object of the data elements identified in the CReq message defined in Table B.3.
- Encrypts the JSON object according to JWE (RFC 7516) using the CEK_{S-A} obtained in Section 6.2.3.3 and JWE Compact Serialization. The parameter values for this version of the specification and to be included in the JWE protected header are:
 - "alg": dir
 - "enc":

- For SDK Type = 01: either A128CBC-HS256 or A128GCM
- For SDK Type = 02: A128GCM
- "kid":ACS Transaction ID
- All other parameters: optional

If the algorithm is A128CBC-HS256 use the full CEK_{S-A} and a fresh 128-bit random data as IV or if the algorithm is A128GCM use the leftmost 128 bits of CEK_{S-A} with SDKCounterStoA (padded to the left with '00' bytes) as the IV.

- Increments SDKCounterStoA.
 - If SDKCounterStoA ≠ zero, sends the resulting JWE to the ACS as the protected CReq message.
 - If SDKCounterStoA = zero, ceases processing and reports error.

6.2.4.2 3DS SDK—CRes

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

- Decrypts the message according to JWE (RFC 7516) using either A128CBC-HS256 or A128GCM and the CEK_{A-S} obtained in Section 6.2.3.3 as identified by the "enc" and "kid" parameters in the protected header. If decryption fails, ceases processing and reports error.
- Checks that ACSCounterAtoS in the decrypted message numerically equals SDKCounterAtoS. If not, ceases processing and reports error.
- Increments SDKCounterAtoS. If SDKCounterAtoS = zero, ceases processing and reports error.

6.2.4.3 ACS—CReq

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

- Decrypts the message according to JWE (RFC 7516) using either A128CBC-HS256 or A128GCM and the CEK_{S-A} obtained in Section 6.2.3.2 as identified by the "enc" and "kid" parameters in the protected header. If decryption fails, ceases processing and reports error.
- Checks that SDKCounterStoA in the decrypted message numerically equals ACSCounterStoA. If not, ceases processing and reports error.
- Increments ACSCounterStoA. If ACSCounterStoA = zero, ceases processing and reports error.

6.2.4.4 ACS—CRes

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

- Creates a JSON object of the data elements identified in the CRes message defined in Table B.4.
- Encrypts the JSON object according to JWE (RFC 7516) using the same "enc" algorithm used by the 3DS SDK for the CReq message, the CEK_{A-S} obtained in Section 6.2.3.2 identified by "kid" and JWE Compact Serialization. The parameter values for this version of the specification and to be included in the JWE protected header are:
 - "alg": dir
 - "enc": either A128CBC-HS256 or A128GCM

- "kid": ACS Transaction ID
- All other parameters: optional

If the algorithm is A128CBC-HS256 use the full CEK_{A-S} and a fresh 128-bit random data as IV or if the algorithm is A128GCM use the rightmost 128 bits of CEK_{A-S} with $ACSCounterAtoS$ (padded to the left with 'FF' bytes) as the IV.

- Increments $ACSCounterAtoS$.
 - If $ACSCounterAtoS \neq$ zero, sends the resulting JWE to the 3DS SDK as the protected CReq message.
 - If $ACSCounterAtoS =$ zero, ceases processing and reports error.

Annex A 3-D Secure Data Elements

This annex contains an alphabetical listing of all EMV 3-D Secure data elements. Data element information and the Standards used to identify the information are as follows:

- Data Element Name—Identifies the data element
- Field Name—Identifies the field name for the data element
- Description—Identifies the data element purpose, and additional detail as applicable
- Source—Identifies the 3-D Secure component that is responsible to provide the data element in the message
- Length/Format/Values—Identifies the value length detail, JSON data format, and if applicable, the values associated with the data element. The term "character" in the Length Edit criteria refers to one UTF-8 character. The length of a JSON object is the length in characters of the overall string representing the object.
- Device Channel—Identifies the inclusion of a data element in a message based on the Device Channel used for a specific transaction. The following Standard is used to identify the Device Channel type:
 - **01-APP**—App-based Authentication
 - **02-BRW**—Browser-based Authentication
 - **03-3RI**—Deviceless Payment Authentication/Verification of Account
- Message Category—Identifies the inclusion of a data element in a Message based on the type of transaction. The following Standard is used to identify the Authentication type:
 - **01-PA**—Payment Authentication
 - **02-NPA**—Non-Payment Authentication
- Message Inclusion—Identifies the Message Type(s) that the data element is included in, and whether the inclusion of the data element in the Message Type is Required, Optional, or Conditional for both Message Categories. Unless explicitly noted otherwise in Table A.1, if a field is required for the Device Channel and Message Category of a specific transaction, the value must be present and not be empty or null.

The following Standards are used to identify Inclusion values:

- **R** = Required—Sender shall include the data element in the identified Message Type, Device Channel, and Message Category; Recipient shall check for data element presence and Validate data element contents.
- **C** = Conditional—Sender shall include the data element in the identified Message Type if the Conditional Inclusion requirements are met; Recipient shall check for data element presence and Validate data element contents. When no data is to be sent for an Optional data element (including a Conditional data element that is not required based on the contents of the message), the data element should be absent.
- **O** = Optional—Sender may include the data element in the identified Message Type; Recipient shall Validate the data element contents when present. When no data is to be sent for an Optional data element (including a Conditional data element that is not required based on the contents of the message), the data element should be absent.
- Conditional Inclusion—Identifies the applicable conditions for including the data element in the applicable Message Type with the responsibility of the Source component to meet the conditions.

Example:

Data Element/Field Name	Description	Source	Length/Format/Values	Device Channel	Message Category	Message Inclusion	Conditional Inclusion
3DS Requestor URL Field Name: threeDSRequestorURL	Fully Qualified URL of 3DS Requestor website or customer care site. This data element provides additional information to the receiving 3-D Secure system if a problem arises and should provide contact information.	3DS Server	Length: Variable, maximum 2048 characters JSON Data Type: String Value accepted: <ul style="list-style-type: none">● Fully Qualified URL Example:<ul style="list-style-type: none">● https://server.domainname.com	01-APP 02-BRW 03-3RI	01-PA 02-NPA	AReq = R	

In the example above, the 3DS Requestor URL is provided by the 3DS Server and is used for both App-based and Browser-based Authentication in the Authentication Request (AReq) message.

A.1 Missing Required Fields

A data field is missing if either:

- the name/value pair is absent, or
- the field name is present but the value is empty or null

Unless explicitly noted, if a required field is missing, the receiving component returns an Error Message as defined in Section A.9 with the applicable Error Component and Error Code = 201. This applies whether the field is always Required or Conditionally required.

A.2 Field Edit Criteria

Only the specified validations are to be performed. Do not reject a message based on any validation that is not listed in the tables in this annex. If a field is present, but its value does not conform to the edit criteria specified in Table A.1, the receiving component returns an Error Message as defined in Section A.9 with the applicable Error Component and Error Code = 203.

A.3 Encryption of AReq Data

The AReq message contains fields that are included without encryption (these are protected in transit by the secure links defined in Section 6.1). Additionally, the Device Information data element is encrypted by the 3DS SDK and passed to the 3DS Server before inclusion in the message to the DS. All data that is to be encrypted is sent as one block of encrypted data in the 3DS SDK Encrypted Data field as a JWE object. Only the 3DS SDK and DS process the JWE object before being broken down into its constituent fields. The resulting decrypted data will be placed into the AReq message to the ACS unencrypted in the Device Information data element.

A.4 EMV 3-D Secure Data Elements

Table A.1: EMV 3-D Secure Data Elements

Data Element/ Field Name	Description	Source	Length/Format/Values	Device Channel	Message Category	Message Inclusion	Conditional Inclusion
3DS Method Completion Indicator Field Name: threeDSCompInd	Indicates whether the 3DS Method successfully completed.	3DS Server	Length: 1 character JSON Data Type: String Values accepted: <ul style="list-style-type: none">• Y = Successfully completed• N = Did not run or did not successfully complete• U = Unavailable—3DS Method URL was not present in the PRes message data for the card range associated with the Cardholder Account Number.	02-BRW	01-PA 02-NPA	AReq = R	

Data Element/ Field Name	Description	Source	Length/Format/Values	Device Channel	Message Category	Message Inclusion	Conditional Inclusion
3DS Method ID Field Name: threeDSMethodId	Contains the 3DS Server Transaction ID used during the previous execution of the 3DS method.	3DS Server	Length: 36 characters JSON Data Type: String Value accepted: Canonical format as defined in IETF RFC 4122. May utilise any of the specified versions if the output meets specified requirements.	02-BRW	01-PA 02-NPA	AReq = C	Required if 3DS Requestor reuses previous 3DS Method execution.
3DS Requestor App URL Field Name: threeDSRequestorAppURL	3DS Requestor App declaring its URL within the CReq message so that the Authentication app can call the 3DS Requestor App after OOB authentication has occurred. Note: When providing the 3DS Requestor App URL, the 3DS Requestor needs to properly register the URL with the Operating System.	3DS SDK	Length: Variable, maximum 2048 characters JSON Data Type: String Value accepted: <ul style="list-style-type: none">• Universal App Link Example value: https://appname.com Refer to Table 1.3 for Universal App Link definition.	01-APP	01-PA 02-NPA	CReq = C	Required in all CReq messages if 3DS Requestor App URL is provided by the 3DS SDK.

Data Element/ Field Name	Description	Source	Length/Format/Values	Device Channel	Message Category	Message Inclusion	Conditional Inclusion
3DS Requestor App URL Indicator Field Name: threeDSRequestorAppUR LInd	Indicates whether the OOB Authentication App used by the ACS during a challenge supports the 3DS Requestor App URL.	ACS	Length: 1 character JSON Data Type: String Values accepted: <ul style="list-style-type: none">• Y = 3DS Requestor App URL is supported by the OOB Authentication App• N = 3DS Requestor App URL is NOT supported by the OOB Authentication App	01-APP	01-PA 02-NPA	ARes = R	

Data Element/ Field Name	Description	Source	Length/Format/Values	Device Channel	Message Category	Message Inclusion	Conditional Inclusion
3DS Requestor Authentication Indicator Field Name: threeDSRequestorAuthenticationInd	<p>Indicates the type of Authentication request.</p> <p>This data element provides additional information to the ACS to determine the best approach for handling an authentication request.</p>	3DS Server	<p>Length: 2 characters</p> <p>JSON Data Type: String</p> <p>Values accepted:</p> <ul style="list-style-type: none"> • 01 = Payment transaction • 02 = Recurring transaction • 03 = Instalment transaction • 04 = Add card • 05 = Maintain card • 06 = Cardholder verification as part of EMV token ID&V • 07 = Billing Agreement • 08 = Split shipment • 09 = Delayed shipment • 10 = Split payment • 11–79 = Reserved for EMVCo future use (values invalid until defined by EMVCo) • 80–99 = Reserved for DS use 	01-APP 02-BRW	01-PA 02-NPA	AReq = R	

Data Element/ Field Name	Description	Source	Length/Format/Values	Device Channel	Message Category	Message Inclusion	Conditional Inclusion
3DS Requestor Authentication Information Field Name: threeDSRequestorAuthenticationInfo	Information about how the 3DS Requestor authenticated the Cardholder before or during the transaction.	3DS Server	Length: 1–3 elements JSON Data Type: Array of Objects Refer to Table A.12 for data elements to include. Note: Data will be formatted into a JSON Array of Objects prior to being placed into the 3DS Requestor Authentication Information field of the message.	01-APP 02-BRW	01-PA 02-NPA	AReq = O	Optional, recommended to include.
3DS Requestor Authentication Method Verification Indicator Field Name: threeDSReqAuthMethodInd	Value that represents the signature verification performed by the DS on the mechanism (e.g., FIDO) used by the cardholder to authenticate to the 3DS Requestor.	DS	Length: 2 characters JSON Data Type: String Values accepted: <ul style="list-style-type: none">• 01 = Verified• 02 = Failed• 03 = Not Performed• 04–79 = Reserved for EMVCo future use (values invalid until defined by EMVCo)• 80–99 = Reserved for DS use	01-APP 02-BRW 03-3RI	01-PA 02-NPA	AReq = C	Conditional based on DS rules. The DS populates the AReq with this data element prior to passing to the ACS.

3DS Requestor Challenge Indicator Field Name: threeDSRequestorChallengeInd	Indicates whether a challenge is requested for this transaction. Example: For 01-PA, a 3DS Requestor may have concerns about the transaction, and request a challenge. For 02-NPA, a challenge may be necessary when adding a new card to a wallet. Note: When providing two preferences, the 3DS Requestor ensures that they are in preference order and are not conflicting. For example, 02 = No challenge requested and 04 = Challenge requested (Mandate).	3DS Server DS	Length: 1–2 elements JSON Data Type: Array of String String: 2 characters Values accepted: <ul style="list-style-type: none">• 01 = No preference• 02 = No challenge requested• 03 = Challenge requested (3DS Requestor preference)• 04 = Challenge requested (Mandate)• 05 = No challenge requested (transactional risk analysis is already performed)• 06 = No challenge requested (Data share only)• 07 = No challenge requested (strong consumer authentication is already performed)• 08 = No challenge requested (utilise Trust List exemption if no challenge required)• 09 = Challenge requested (Trust List)	01-APP 02-BRW 03-3RI	01-PA 02-NPA	AReq = O	
--	---	---------------	--	----------------------------	-----------------	----------	--

Data Element/ Field Name	Description	Source	Length/Format/Values	Device Channel	Message Category	Message Inclusion	Conditional Inclusion
			<p>prompt requested if challenge required)</p> <ul style="list-style-type: none"> • 10 = No challenge requested (utilise low value exemption) • 11 = No challenge requested (Secure corporate payment exemption) • 12 = Challenge requested (Device Binding prompt requested if challenge required) • 13 = Challenge requested (Issuer requested) • 14 = Challenge requested (Merchant initiated transactions) • 15–79 = Reserved for EMVCo future use (values invalid until defined by EMVCo) • 80–99 = Reserved for DS use <p>Note: If the element is not provided, the expected action is that the ACS would interpret as 01 = No preference.</p>				

Data Element/ Field Name	Description	Source	Length/Format/Values	Device Channel	Message Category	Message Inclusion	Conditional Inclusion
3DS Requestor Decoupled Max Time Field Name: threeDSRequestorDecMa xTime	Indicates the maximum amount of time that the 3DS Requestor will wait for an ACS to provide the results of a Decoupled Authentication transaction (in minutes).	3DS Server	Length: 5 characters JSON Data Type: String Numeric values between 00001 and 10080 accepted.	01-APP 02-BRW 03-3RI	01-PA 02-NPA	AReq = C	Required if 3DS Requestor Decoupled Request Indicator = Y or F or B.

3DS Requestor Decoupled Request Indicator Field Name: threeDSRequestorDecRe qInd	Indicates whether the 3DS Requestor requests the ACS to utilise Decoupled Authentication and agrees to utilise Decoupled Authentication if the ACS confirms its use. Note: if the element is not provided, the expected action is for the ACS to interpret as N (Do not use Decoupled Authentication).	3DS Server	Length: 1 character JSON Data Type: String Values accepted: <ul style="list-style-type: none">• Y = Decoupled Authentication is supported and is preferred as a primary challenge method if a challenge is necessary (Transaction Status = D in ARes).• N = Do not use Decoupled Authentication.• F = Decoupled Authentication is supported and is to be used only as a fallback challenge method if a challenge is necessary (Transaction Status = D in RReq).• B = Decoupled Authentication is supported and can be used as a primary or fallback challenge method if a challenge is necessary (Transaction Status = D in either ARes or RReq).	01-APP 02-BRW 03-3RI	01-PA 02-NPA	AReq = O	
--	---	------------	--	----------------------------	-----------------	----------	--

Data Element/ Field Name	Description	Source	Length/Format/Values	Device Channel	Message Category	Message Inclusion	Conditional Inclusion
3DS Requestor ID Field Name: threeDSRequestorID	DS-defined 3DS Requestor identifier.	3DS Server	Length: Variable, maximum 35 characters JSON Data Type: String Values accepted: <ul style="list-style-type: none">• Any individual DS may impose specific formatting, character and/or other requirements on the contents of this field.	01-APP 02-BRW 03-3RI	01-PA 02-NPA	AReq = R	
3DS Requestor Name Field Name: threeDSRequestorName	DS-defined 3DS Requestor name.	3DS Server	Length: Variable, maximum 40 characters JSON Data Type: String Values accepted: <ul style="list-style-type: none">• Any individual DS may impose specific formatting, character and/or other requirements on the contents of this field.	01-APP 02-BRW 03-3RI	01-PA 02-NPA	AReq = R	

Data Element/ Field Name	Description	Source	Length/Format/Values	Device Channel	Message Category	Message Inclusion	Conditional Inclusion
3DS Requestor Prior Transaction Authentication Information Field Name: threeDSRequestorPriorAuthenticationInfo	Information about how the 3DS Requestor authenticated the cardholder as part of a previous 3DS transaction.	3DS Server	Length: Variable, 1–3 elements JSON Data Type: Array of Objects Refer to Table A.13 for data elements to include. Note: Data will be formatted into a JSON object prior to being placed into the 3DS Requestor Prior Transaction Authentication Information field of the message.	01-APP 02-BRW 03-3RI	01-PA 02-NPA	AReq = C	Required for 3RI in the case of Decoupled Authentication Fallback or for SPC
3DS Requestor SPC Support Field Name: threeDSRequestorSpcSupport	Indicate if the 3DS Requestor supports the SPC authentication. Note: If present, this field contains the value Y.	3DS Server	JSON Data Type: String Value accepted: • Y = Supported	02-BRW	01-PA 02-NPA	AReq = C	Required if supported by the 3DS Requestor

Data Element/ Field Name	Description	Source	Length/Format/Values	Device Channel	Message Category	Message Inclusion	Conditional Inclusion
3DS Requestor URL Field Name: threeDSRequestorURL	<p>The Fully Qualified URL of the 3DS Requestor website or customer care site.</p> <p>This data element provides additional information to the receiving 3-D Secure system if a problem arises and contact information should be provided.</p>	3DS Server	<p>Length: Variable, maximum 2048 characters JSON Data Type: String Value accepted:</p> <ul style="list-style-type: none"> • Fully Qualified URL <p>Example:</p> <ul style="list-style-type: none"> • https://server.domainname.com 	01-APP 02-BRW 03-3RI	01-PA 02-NPA	AReq = R	
3DS Server Operator ID Field Name: threeDSServerOperatorID	<p>DS-assigned 3DS Server identifier.</p> <p>Each DS can provide a unique ID to each 3DS Server on an individual basis.</p>	3DS Server	<p>Length: Variable, maximum 32 characters JSON Data Type: String Value accepted:</p> <ul style="list-style-type: none"> • Any individual DS may impose specific formatting and character requirements on the contents of this field. 	01-APP 02-BRW 03-3RI	01-PA 02-NPA	AReq = C PReq = C	Requirements for the presence of this field are DS-specific.
3DS Server Reference Number Field Name: threeDSServerRefNumber	Unique identifier assigned by the EMVCo secretariat upon testing and approval.	3DS Server	<p>Length: Variable, maximum 32 characters JSON Data Type: String Value accepted:</p> <ul style="list-style-type: none"> • Set by the EMVCo Secretariat. 	01-APP 02-BRW 03-3RI	01-PA 02-NPA	AReq = R PReq = R ORes = C	Required in ORes message for a 3DS Server receiving an OReq message.

Data Element/ Field Name	Description	Source	Length/Format/Values	Device Channel	Message Category	Message Inclusion	Conditional Inclusion
3DS Server Transaction ID Field Name: threeDSServerTransID	Universally unique transaction identifier assigned by the 3DS Server to identify a single transaction.	3DS Server	Length: 36 characters JSON Data Type: String Value accepted: <ul style="list-style-type: none">• Canonical format as defined in IETF RFC 4122. May utilise any of the specified versions if the output meets specified requirements.	01-APP 02-BRW 03-3RI	01-PA 02-NPA	AReq = R ARes = R CReq = R CRes = R ORes = C PReq = R PRes = R RReq = R RRes = R Erro = C	<ul style="list-style-type: none"> • Required in Error Message if available (e.g., can be obtained from a message or is being generated). • Required in ORes message for a 3DS Server receiving an OReq message.
3DS Server URL Field Name: threeDSServerURL	Fully Qualified URL of the 3DS Server to which the DS will send the RReq message after the challenge has completed. Incorrect formatting will result in a failure to deliver the transaction results via the RReq message.	3DS Server	Length: Variable, maximum 2048 characters JSON Data Type: String Value accepted: <ul style="list-style-type: none">• Fully Qualified URL Example value: https://server.adomainname.net	01-APP 02-BRW 03-3RI	01-PA 02-NPA	AReq = R	

3RI Indicator Field Name: threeRIInd	Indicates the type of 3RI request. This data element provides additional information to the ACS to determine the best approach for handling a 3RI request.	3DS Server	Length: 2 characters JSON Data Type: String Values accepted: <ul style="list-style-type: none">• 01 = Recurring transaction• 02 = Instalment transaction• 03 = Add card• 04 = Maintain card information• 05 = Account verification• 06 = Split shipment• 07 = Top-up• 08 = Mail Order• 09 = Telephone Order• 10 = Trust List status check• 11 = Other payment• 12 = Billing Agreement• 13 = Device Binding status check• 14 = Card Security Code status check• 15 = Delayed shipment• 16 = Split payment• 17–79 = Reserved for EMVCo future use	03-3RI	01-PA 02-NPA	AReq = R	
---	---	------------	---	--------	-----------------	----------	--

Data Element/ Field Name	Description	Source	Length/Format/Values	Device Channel	Message Category	Message Inclusion	Conditional Inclusion
			<p>(values invalid until defined by EMVCo)</p> <ul style="list-style-type: none"> • 80–99 = Reserved for DS use 				
Accept Language Field Name: acceptLanguage	Value representing the Browser language preference present in the HTTP header, as defined in IETF BCP 47.	3DS Server	<p>Size: Variable, 1–99 elements</p> <p>JSON Data Type: Array of String</p> <p>String: Variable, maximum 100 characters</p>	02-BRW	01-PA 02-NPA	AReq = R	
Account Type Field Name: acctType	Indicates the type of account. For example, for a multi-account card product.	3DS Server	<p>Length: 2 characters</p> <p>JSON Data Type: String</p> <p>Values accepted:</p> <ul style="list-style-type: none"> • 01 = Not Applicable • 02 = Credit • 03 = Debit • 04–79 = Reserved for EMVCo future use (values invalid until defined by EMVCo) • 80–99 = DS or Payment System-specific 	01-APP 02-BRW 03-3RI	01-PA 02-NPA	AReq = C	<p>Required if 3DS Requestor is asking Cardholder which Account Type they are using before making the purchase.</p> <p>Required in some markets (for example, for Merchants in Brazil). Otherwise, Optional.</p>

Data Element/ Field Name	Description	Source	Length/Format/Values	Device Channel	Message Category	Message Inclusion	Conditional Inclusion
Acquirer BIN Field Name: acquirerBIN	Acquiring institution identification code as assigned by the DS receiving the AReq message.	3DS Server	Length: Variable, maximum 11 characters JSON Data Type: String Value accepted: This value correlates to the Acquirer BIN as defined by each Payment System or DS.	01-APP 02-BRW 03-3RI	01-PA 02-NPA	01-PA: AReq = R 02-NPA: AReq = O	
Acquirer Country Code Field Name: acquirerCountryCode	The code of the country where the acquiring institution is located (in accordance with ISO 3166-1). The DS may edit the value provided by the 3DS Server.	3DS Server DS	Length: 3 characters JSON Data Type: String Values accepted: • ISO 3166-1 numeric three-digit country codes, other than exceptions listed in Table A.5.	01-APP 02-BRW 03-3RI	01-PA 02-NPA	AReq = R	

Data Element/ Field Name	Description	Source	Length/Format/Values	Device Channel	Message Category	Message Inclusion	Conditional Inclusion
Acquirer Country Code Source Field Name: acquirerCountryCodeSo urce	This data element is populated by the system setting the Acquirer Country Code. The DS may edit the value provided by the 3DS Server.	3DS Server DS	Length: 2 characters JSON Data Type: String Values accepted: <ul style="list-style-type: none">• 01 = 3DS Server• 02 = DS• 03-79 = Reserved for EMVCo future use (values invalid until defined by EMVCo)• 80-99 = Reserved for DS use	01-APP 02-BRW 03-3RI	01-PA 02-NPA	AReq = R	
Acquirer Merchant ID Field Name: acquirerMerchantID	Acquirer-assigned Merchant identifier. This may be the same value that is used in authorisation requests sent on behalf of the 3DS Requestor and is represented in ISO 8583-1 formatting requirements.	3DS Server	Length: Variable, maximum 35 characters JSON Data Type: String Value accepted: <ul style="list-style-type: none">• Individual Directory Servers may impose specific format and character requirements on the contents of this field.	01-APP 02-BRW 03-3RI	01-PA 02-NPA	01-PA: AReq = R 02-NPA AReq = O	

Data Element/ Field Name	Description	Source	Length/Format/Values	Device Channel	Message Category	Message Inclusion	Conditional Inclusion
ACS Challenge Mandated Indicator Field Name: acsChallengeMandated	Indication of whether a challenge is required for the transaction to be authorised due to local/regional mandates or other variable.	ACS	Length: 1 character JSON Data Type: String Values accepted: <ul style="list-style-type: none">• Y = Challenge is mandated• N = Challenge is not mandated	01-APP 02-BRW 03-3RI	01-PA 02-NPA	ARes = C	Required if Transaction Status = C or D.
ACS Counter ACS to SDK Field Name: acsCounterAtoS	Counter used as a security measure in the ACS to 3DS SDK secure channel. Note: The counter is the decimal value equivalent of the byte, encoded as a numeric string.	ACS	Length: 3 characters JSON Data Type: String Values accepted: <ul style="list-style-type: none">• 000–255	01-APP	01-PA 02-NPA	CRes = R	

Data Element/ Field Name	Description	Source	Length/Format/Values	Device Channel	Message Category	Message Inclusion	Conditional Inclusion
ACS Decoupled Confirmation Indicator Field Name: acsDecConInd	Indicates whether the ACS confirms utilisation of Decoupled Authentication and agrees to utilise Decoupled Authentication to authenticate the Cardholder.	ACS	<p>Length: 1 character JSON Data Type: String Values accepted:</p> <ul style="list-style-type: none"> • Y = Confirms Decoupled Authentication will be utilised • N = Decoupled Authentication will not be utilised <p>Note: if 3DS Requestor Decoupled Request Indicator = N, a value of Y cannot be returned in the ACS Decoupled Confirmation Indicator.</p> <p>Note: if Transaction Status = D, a value of N is not valid.</p>	01-APP 02-BRW 03-3RI	01-PA 02-NPA	ARes = C	Required if Transaction Status = D
ACS Ephemeral Public Key (QT) Field Name: acsEphemPubKey	<p>Public key component of the ephemeral key pair generated by the ACS and used to establish session keys between the 3DS SDK and the ACS.</p> <p>See Section 6.2.3.2 for additional detail.</p>	ACS	<p>Length: Variable, maximum 256 characters JSON Data Type: Object</p>	01-APP	01-PA 02-NPA	See ACS Signed Content	See ACS Signed Content.

Data Element/ Field Name	Description	Source	Length/Format/Values	Device Channel	Message Category	Message Inclusion	Conditional Inclusion
ACS HTML Field Name: acsHTML	HTML provided by the ACS in the CRes message. Utilised when HTML is specified in the ACS UI Type during the Cardholder challenge.	ACS	<p>Length: Variable, maximum 300 kB</p> <p>JSON Data Type: String</p> <p>Value accepted:</p> <ul style="list-style-type: none"> • Base64url-encoded HTML <p>This value will be Base64url-encoded prior to being placed into the CRes message.</p>	01-APP	01-PA 02-NPA	CRes = C	Required if ACS UI Type = 05 or 06.
ACS Operator ID Field Name: acsOperatorID	<p>DS assigned ACS identifier.</p> <p>Each DS can provide a unique ID to each ACS on an individual basis.</p>	ACS	<p>Length: Variable, maximum 32 characters</p> <p>JSON Data Type: String</p> <p>Value accepted:</p> <ul style="list-style-type: none"> • Any individual DS may impose specific formatting and character requirements on the contents of this field. 	01-APP 02-BRW 03-3RI	01-PA 02-NPA	ARes = C	Requirements for the presence of this field are DS specific.
ACS Reference Number Field Name: acsReferenceNumber	Unique identifier assigned by the EMVCo Secretariat upon Testing and Approval.	ACS	<p>Length: Variable, maximum 32 characters</p> <p>JSON Data Type: String</p> <p>Value accepted:</p> <ul style="list-style-type: none"> • Set by the EMVCo Secretariat. 	01-APP 02-BRW 03-3RI	01-PA 02-NPA	ARes = R ORes = C	Required in ORes message for an ACS receiving an OReq message.

Data Element/ Field Name	Description	Source	Length/Format/Values	Device Channel	Message Category	Message Inclusion	Conditional Inclusion
ACS Rendering Type Field Name: acsRenderingType	Identifies the ACS Interface and ACS UI Template that the ACS will first present to the consumer.	ACS	<p>JSON Data Type: Object Values accepted:</p> <ul style="list-style-type: none"> Refer to Table A.14 for data elements to include. <p>Note: Data will be formatted into a JSON object prior to being placed into the acsRenderingType field of the message.</p>	01-APP	01-PA 02-NPA	ARes = C RReq = C	<ul style="list-style-type: none"> For ARes, required if Transaction Status = C. For RReq, required unless ACS Decoupled Confirmation Indicator = Y.
ACS Signed Content Field Name: acsSignedContent	<p>Contains the JWS object (represented as a string) created by the ACS for the ARes message.</p> <p>See Section 6.2.3.2 for details.</p>	ACS	<p>Length: Variable JSON Data Type: String Values accepted:</p> <ul style="list-style-type: none"> The body of JWS object (represented as a string) will contain the following data elements as defined in Table A.1: <ul style="list-style-type: none"> ACS URL ACS Ephemeral Public Key (Q_T) SDK Ephemeral Public Key (QC) 	01-APP	01-PA 02-NPA	ARes = C	Required if the Transaction Status = C.

Data Element/ Field Name	Description	Source	Length/Format/Values	Device Channel	Message Category	Message Inclusion	Conditional Inclusion
ACS Transaction ID Field Name: acsTransID	Universally Unique transaction identifier assigned by the ACS to identify a single transaction.	ACS	Length: 36 characters JSON Data Type: String Value accepted: <ul style="list-style-type: none"> Canonical format as defined in IETF RFC 4122. May utilise any of the specified versions if the output meets specified requirements. 	01-APP 02-BRW 03-3RI	01-PA 02-NPA	ARes = R CReq = R CRes = R ORes = C RReq = R RRes = R Erro = C	<ul style="list-style-type: none"> Required in Error Message if available (e.g., can be obtained from a message or is being generated). Required in ORes message for an ACS receiving an OReq message.

Data Element/ Field Name	Description	Source	Length/Format/Values	Device Channel	Message Category	Message Inclusion	Conditional Inclusion
ACS UI Type Field Name: acsUiType	User interface type that the 3DS SDK will render, which includes the specific data mapping and requirements.	ACS	Length: 2 characters JSON Data Type: String Values accepted: <ul style="list-style-type: none"> • 01 = Text • 02 = Single Select • 03 = Multi Select • 04 = OOB • 05 = HTML • 06 = HTML OOB • 07 = Information • 08–79 = Reserved for EMVCo future use (values invalid until defined by EMVCo) • 80–99 = Reserved for DS use 	01-APP	01-PA 02-NPA	CRes = C	Required except for Final CRes message.

Data Element/ Field Name	Description	Source	Length/Format/Values	Device Channel	Message Category	Message Inclusion	Conditional Inclusion
ACS URL Field Name: acsURL	<p>Fully Qualified URL of the ACS to be used for the challenge.</p> <p>01-APP—3DS SDK will send the Challenge Request to this URL</p> <p>02-BRW—3DS Requestor will post the CReq to this URL via the challenge iframe.</p> <p>For App-based, this data element is contained within the ACS Signed Content JWS Object.</p> <p>For Browser-based, this data element is present as its own object.</p>	ACS	<p>Length: Variable, maximum 2048 characters</p> <p>JSON Data Type: String</p> <p>Value accepted:</p> <ul style="list-style-type: none"> • Fully Qualified URL. <p>Example:</p> <p>https://server.acsdomainname.com</p>	01-APP 02-BRW	01-PA 02-NPA	01-APP: see ACS Signed Content 02-BRW: ARes = C	<ul style="list-style-type: none"> • For 01-APP, see ACS Signed Content. • For 02-BRW, required if Transaction Status = C.
Address Match Indicator Field Name: addrMatch	Indicates whether the Cardholder Shipping Address and Cardholder Billing Address are the same.	3DS Server	<p>Length: 1 character</p> <p>JSON Data Type: String</p> <p>Values accepted:</p> <ul style="list-style-type: none"> • Y = Shipping Address matches Billing Address • N = Shipping Address does not match Billing Address 	01-APP 02-BRW	01-PA 02-NPA	AReq = O	

Data Element/ Field Name	Description	Source	Length/Format/Values	Device Channel	Message Category	Message Inclusion	Conditional Inclusion
App IP Address Field Name: appIp	External IP address (i.e., the device public IP address) used by the 3DS Requestor App when it connects to the 3DS Requestor environment.	3DS Server	Length: Variable, maximum 45 characters JSON Data Type: String Values accepted: <ul style="list-style-type: none">• IPv4 address. Refer to RFC 791.• IPv6 address. Refer to RFC 4291.	01-APP	01-PA 02-NPA	AReq = C	Required unless market or regional mandate restricts sending this information.

<p>Authentication Method</p> <p>Field Name: authenticationMethod</p>	<p>Indicates the list of authentication types the Issuer will use to challenge the Cardholder, when in the ARes message or what was used by the ACS when in the RReq message.</p> <p>Note: For 03-3RI, only present for Decoupled Authentication.</p>	ACS	<p>Size: Variable, 1–99 elements</p> <p>JSON Data Type: Array of String.</p> <p>String: 2 characters</p> <p>Values accepted:</p> <ul style="list-style-type: none"> • 01 = Static Passcode • 02 = SMS OTP • 03 = Key fob or EMV card reader OTP • 04 = App OTP • 05 = OTP Other • 06 = KBA • 07 = OOB Biometrics • 08 = OOB Login • 09 = OOB Other • 10 = Other • 11 = Push Confirmation • 12 = Decoupled • 13 = WebAuthn • 14 = SPC • 15 = Behavioural biometrics • 16–79 = Reserved for future EMVCo use (values invalid until defined by EMVCo) 	01-APP 02-BRW 03-3RI	01-PA 02-NPA	ARes = C RReq = C	<ul style="list-style-type: none"> • Required in the ARes message if Transaction Status = C or D. • Required in the RReq message if Transaction Status = Y or N.
--	---	-----	--	----------------------------	-----------------	----------------------	--

Data Element/ Field Name	Description	Source	Length/Format/Values	Device Channel	Message Category	Message Inclusion	Conditional Inclusion
			<ul style="list-style-type: none"> 80–99 = Reserved for DS use <p>If SDK Type = 02 and Split-SDK Type/Limited Indicator = Y, a value of 01 or 06 is not valid.</p>				
Authentication Value Field Name: authenticationValue	<p>Payment System-specific value provided by the ACS or the DS using an algorithm defined by Payment System.</p> <p>Authentication Value may be used to provide proof of authentication.</p>	ACS DS	<p>Length: Variable, maximum 4000 characters. Actual length defined by Payment System rules.</p> <p>JSON Data Type: String</p> <p>Example:</p> <p>A 20-byte value that has been Base64 encoded, giving a 28-byte result.</p>	01-APP 02-BRW 03-3RI	01-PA 02-NPA	ARes = C RReq = C	<p>01-PA: Required if Transaction Status = Y or A</p> <p>Conditional based on DS rules if Transaction Status = I</p> <p>Omitted from the RReq message when sent as an abandonment notification</p> <p>02-NPA: Conditional based on DS rules.</p>
Broadcast Information Field Name: broadInfo	Structured information sent between the 3DS Server, the DS and the ACS.	3DS Server DS ACS	<p>Length: Variable, maximum 4096 characters</p> <p>JSON Data Type: Object</p> <p>Refer to Table A.27 for data elements to include.</p>	01-APP 02-BRW 03-3RI	01-PA 02-NPA	AReq = O ARes = O	

Data Element/ Field Name	Description	Source	Length/Format/Values	Device Channel	Message Category	Message Inclusion	Conditional Inclusion
Browser Accept Headers Field Name: browserAcceptHeader	Exact content of the HTTP accept headers as sent to the 3DS Requestor from the Cardholder Browser.	3DS Server	Length: Variable, maximum 2048 characters JSON Data Type: String Value accepted: <ul style="list-style-type: none">• If the total length of the accept header sent by the Browser exceeds 2048 characters, the 3DS Server truncates the excess portion. Refer to Section A.6 for additional detail.	02-BRW	01-PA 02-NPA	AReq = R	
Browser IP Address Field Name: browserIP	IP address of the Browser as returned by the HTTP headers to the 3DS Requestor.	3DS Server	Length: Variable, maximum 45 characters JSON Data Type: String Values accepted: <ul style="list-style-type: none">• IPv4 address. Refer to RFC 791.• IPv6 address. Refer to RFC 4291.	02-BRW	01-PA 02-NPA	AReq = C	Required unless market or regional mandate restricts sending this information.

Data Element/ Field Name	Description	Source	Length/Format/Values	Device Channel	Message Category	Message Inclusion	Conditional Inclusion
Browser Java Enabled Field Name: browserJavaEnabled	Boolean that represents the ability of the Cardholder Browser to execute Java. Value is returned from the navigator.javaEnabled property. Refer to Section A.6 for additional detail.	3DS Server	JSON Data Type: Boolean Values accepted: <ul style="list-style-type: none">• true• false	02-BRW	01-PA 02-NPA	AReq = C	Required when Browser JavaScript Enabled = true; otherwise Optional.
Browser JavaScript Enabled Field Name: browserJavascriptEnabled	Boolean that represents the ability of the Cardholder Browser to execute JavaScript. Refer to Section A.6 for additional detail.	3DS Server	JSON Data Type: Boolean Values accepted: <ul style="list-style-type: none">• true• false	02-BRW	01-PA 02-NPA	AReq = R	
Browser Language Field Name: browserLanguage	Value representing the Browser language as defined in IETF BCP47. Returned from navigator.language property. Refer to Section A.6 for additional detail.	3DS Server	Length: Variable, Maximum 35 characters JSON Data Type: String	02-BRW	01-PA 02-NPA	AReq = C	Required when Browser JavaScript Enabled = true; otherwise Optional.

Data Element/ Field Name	Description	Source	Length/Format/Values	Device Channel	Message Category	Message Inclusion	Conditional Inclusion
Browser Screen Color Depth Field Name: browserColorDepth	<p>Value representing the bit depth of the colour palette for displaying images, in bits per pixel.</p> <p>Obtained from the Cardholder Browser using the screen.colorDepth property.</p> <p>Refer to Section A.6 for more details.</p>	3DS Server	<p>Length: 1–2 characters; numeric</p> <p>JSON Data Type: String</p> <p>Values accepted:</p> <ul style="list-style-type: none"> • 1–99 <p>For a list of possible values, refer to https://www.w3schools.com/jsref/prop_screen_color_depth.asp</p> <p>Note: If an ACS does not support the value provided, then the ACS can use the closest supported value. For example, if the value provided = 30 and the ACS does not support that value, then the ACS could use the value = 24.</p>	02-BRW	01-PA 02-NPA	AReq = C	Required when Browser JavaScript Enabled = true; otherwise Optional.
Browser Screen Height Field Name: browserScreenHeight	<p>Total height of the Cardholder's screen in pixels.</p> <p>Value is returned from the screen.height property.</p> <p>Refer to Section A.6 for additional detail.</p>	3DS Server	<p>Length: Variable, 1–6 characters; numeric</p> <p>JSON Data Type: String</p>	02-BRW	01-PA 02-NPA	AReq = C	Required when Browser JavaScript Enabled = true; otherwise Optional.

Data Element/ Field Name	Description	Source	Length/Format/Values	Device Channel	Message Category	Message Inclusion	Conditional Inclusion
Browser Screen Width Field Name: browserScreenWidth	Total width of the Cardholder's screen in pixels. Value is returned from the screen.width property. Refer to Section A.6 for additional detail.	3DS Server	Length: Variable, 1–6 characters; numeric JSON Data Type: String	02-BRW	01-PA 02-NPA	AReq = C	Required when Browser JavaScript Enabled = true; otherwise Optional.
Browser Time Zone Field Name: browserTZ	Time zone offset in minutes between UTC and the Cardholder Browser local time. Note that the offset is positive if the local time zone is behind UTC and negative if it is ahead.	3DS Server	Length: Variable, 1–5 characters JSON Data Type: String Value accepted: <ul style="list-style-type: none">• Value is returned from the getTimezoneOffset() method. Example time zone offset values in minutes: If UTC -5 hours: <ul style="list-style-type: none">• 300• +300 If UTC +5 hours: <ul style="list-style-type: none">• -300 Refer to Section A.6 for additional detail.	02-BRW	01-PA 02-NPA	AReq = C	Required when Browser JavaScript Enabled = true; otherwise Optional.

Data Element/ Field Name	Description	Source	Length/Format/Values	Device Channel	Message Category	Message Inclusion	Conditional Inclusion
Browser User-Agent Field Name: browserUserAgent	Exact content of the HTTP user-agent header.	3DS Server	Length: Variable, maximum 2048 characters JSON Data Type: String Value accepted: Note: If the total length of the User-Agent sent by the Browser exceeds 2048 characters, the 3DS Server truncates the excess portion. Refer to Section A.6 for additional detail.	02-BRW	01-PA 02-NPA	AReq = R	
Browser User Device ID Field Name: deviceId	Unique and immutable identifier linked to a device that is consistent across 3DS transactions for the specific user device. Examples: <ul style="list-style-type: none">• Hardware Device ID• Platform-calculated device fingerprint Refer to D021 in the SDK Device Information	3DS Server	Length: Variable, maximum 64 characters JSON Data Type: String	02-BRW	01-PA 02-NPA	AReq = C	Required if available.

Data Element/ Field Name	Description	Source	Length/Format/Values	Device Channel	Message Category	Message Inclusion	Conditional Inclusion
Browser User ID Field Name: userId	<p>Identifier of the transacting user's Browser Account ID.</p> <p>This identifier is a unique immutable hash of the user's account identifier for the given Browser, provided as a string.</p> <p>Note: Cardholders may have more than one account on a given Browser.</p> <p>Refer to D026 in the SDK Device Information</p>	3DS Server	Length: Variable, maximum 64 characters JSON Data Type: String	02-BRW	01-PA 02-NPA	AReq = C	Required if available.

Data Element/ Field Name	Description	Source	Length/Format/Values	Device Channel	Message Category	Message Inclusion	Conditional Inclusion
Card Range Data Field Name: cardRangeData	Card range data from the DS indicating the most recent protocol versions supported by the ACS, and, optionally, the DS that hosts that range, and, if configured, the ACS URL for the 3DS Method. Additionally, it identifies the 3DS features supported by the ACS, such as Trust List or Decoupled Authentication. There may be as many JSON Objects as there are stored card ranges in the DS.	DS	Size: Variable, 1–200,000 elements JSON Data Type: Array of Objects Values accepted: <ul style="list-style-type: none">• See Table A.6 for Card Range Data elements.	N/A	N/A	PRes = C	Required if the Serial Number has changed in the prior PRes message or is absent in the PReq message AND Not present if the Card Range Data File URL is present
Card Range Data Download Indicator Field Name: cardRangeDataDownload Ind	Indicates if the 3DS Server supports Card Range Data from a file. Note: If present, this field contains the value Y.	3DS Server	Length: 1 character JSON Data Type: String Value accepted: <ul style="list-style-type: none">• Y = Download supported	N/A	N/A	PReq = C	Present only if the 3DS Server supports the Card Range Data File download

Data Element/ Field Name	Description	Source	Length/Format/Values	Device Channel	Message Category	Message Inclusion	Conditional Inclusion
Card Range Data File URL Field Name: cardRangeDataFileURL	Fully Qualified URL of the DS File containing the Card Range Data for download.	DS	Length: Variable, maximum 2048 characters JSON Data Type: String Value accepted: <ul style="list-style-type: none">• Fully Qualified URL Example:<ul style="list-style-type: none">• https://server.dsdomainname.com/cardfile.json	N/A	N/A	PRes = C	Present only if the 3DS Server and the DS are using the Card Range Data File download
Card/Token Expiry Date Field Name: cardExpiryDate	Expiry Date of the PAN or token supplied to the 3DS Requestor by the Cardholder.	3DS Server	Length: 4 characters JSON Data Type: String Format accepted: <ul style="list-style-type: none">• YYMM	01-APP 02-BRW 03-3RI	01-PA 02-NPA	AReq = C	The requirements for the presence of this field are DS specific.
Card Security Code Field Name: cardSecurityCode	Three- or four-digit security code printed on the card.	3DS Server	Length: Variable, 3-4 characters, numeric. Action defined by Payment System rules. JSON Data Type: String	01-APP 02-BRW 03-3RI	01-PA 02-NPA	AReq = C	Conditional based on DS rules

Data Element/ Field Name	Description	Source	Length/Format/Values	Device Channel	Message Category	Message Inclusion	Conditional Inclusion
Card Security Code Status Field Name: cardSecurityCodeStatus	Enables the communication of Card Security Code Status between the ACS, the DS and the 3DS Requestor	ACS DS	Length: 1 character JSON Data Type: String Values accepted: <ul style="list-style-type: none">• Y = Validated• N = Failed validation• U = Status unknown, unavailable, or does not apply	01-APP 02-BRW 03-3RI	01-PA 02-NPA	AReq = C ARes = C	Conditional based on DS rules
Card Security Code Status Source Field Name: cardSecurityCodeStatusSource	This data element will be populated by the system setting Card Security Code Status.	ACS DS	Length: 2 characters JSON Data Type: String Values accepted: <ul style="list-style-type: none">• 01 = DS• 02 = ACS• 03–79 = Reserved for EMVCo future use (values invalid until defined by EMVCo)• 80–99 = Reserved for DS use	01-APP 02-BRW 03-3RI	01-PA 02-NPA	AReq = C ARes = C	Required if the Card Security Code Status is present.
Cardholder Account Identifier Field Name: acctID	Additional information about the account optionally provided by the 3DS Requestor.	3DS Server	Length: Variable, maximum 64 characters JSON Data Type: String	01-APP 02-BRW 03-3RI	01-PA 02-NPA	AReq = O	

Data Element/ Field Name	Description	Source	Length/Format/Values	Device Channel	Message Category	Message Inclusion	Conditional Inclusion
Cardholder Account Information Field Name: acctInfo	Additional information about the Cardholder's account provided by the 3DS Requestor.	3DS Server	Length: Variable JSON Data Type: Object Value accepted: <ul style="list-style-type: none">• Refer to Table A.10 for Cardholder Account Information data elements.	01-APP 02-BRW 03-3RI	01-PA 02-NPA	AReq = O	Optional, but strongly recommended to include.
Cardholder Account Number Field Name: acctNumber	Account number that will be used in the authorisation request for payment transactions. May be represented by PAN, Payment Token.	3DS Server	Length: Variable, 13–19 characters JSON Data Type: String Value accepted: <ul style="list-style-type: none">• Format represented ISO 7812.	01-APP 02-BRW 03-3RI	01-PA 02-NPA	AReq = R	
Cardholder Billing Address City Field Name: billAddrCity	The city of the Cardholder billing address associated with the card used for this purchase.	3DS Server	Length: Variable, maximum 50 characters JSON Data Type: String	01-APP 02-BRW 03-3RI	01-PA 02-NPA	AReq = C	01-PA: Required unless market or regional mandate restricts sending this information. 02-NPA: Required (if available) unless market or regional mandate restricts sending this information.

Data Element/ Field Name	Description	Source	Length/Format/Values	Device Channel	Message Category	Message Inclusion	Conditional Inclusion
Cardholder Billing Address Country Field Name: billAddrCountry	The country of the Cardholder billing address associated with the card used for this purchase.	3DS Server	<p>Length: 3 characters JSON Data Type: String Values accepted:</p> <ul style="list-style-type: none"> • ISO 3166-1 numeric three-digit country code, other than exceptions listed in Table A.5. 	01-APP 02-BRW 03-3RI	01-PA 02-NPA	AReq = C	<p>Required if Cardholder Billing Address State is present.</p> <p>01-PA: Required unless market or regional mandate restricts sending this information.</p> <p>02-NPA: Required (if available) unless market or regional mandate restricts sending this information.</p>

Data Element/ Field Name	Description	Source	Length/Format/Values	Device Channel	Message Category	Message Inclusion	Conditional Inclusion
Cardholder Billing Address Line 1 Field Name: billAddrLine1	First line of the street address or equivalent local portion of the Cardholder billing address associated with the card used for this purchase.	3DS Server	Length: Variable, maximum 50 characters JSON Data Type: String	01-APP 02-BRW 03-3RI	01-PA 02-NPA	AReq = C	01-PA: Required unless market or regional mandate restricts sending this information. 02-NPA: Required (if available) unless market or regional mandate restricts sending this information.
Cardholder Billing Address Line 2 Field Name: billAddrLine2	Second line of the street address or equivalent local portion of the Cardholder billing address associated with the card used for this purchase.	3DS Server	Length: Variable, maximum 50 characters JSON Data Type: String	01-APP 02-BRW 03-3RI	01-PA 02-NPA	AReq = C	Required (if available) unless market or regional mandate restricts sending this information.
Cardholder Billing Address Line 3 Field Name: billAddrLine3	Third line of the street address or equivalent local portion of the Cardholder billing address associated with the card used for this purchase.	3DS Server	Length: Variable, maximum 50 characters JSON Data Type: String	01-APP 02-BRW 03-3RI	01-PA 02-NPA	AReq = C	Required (if available) unless market or regional mandate restricts sending this information.

Data Element/ Field Name	Description	Source	Length/Format/Values	Device Channel	Message Category	Message Inclusion	Conditional Inclusion
Cardholder Billing Address Postal Code Field Name: billAddrPostCode	ZIP or other postal code of the Cardholder billing address associated with the card used for this purchase.	3DS Server	Length: Variable, maximum 16 characters JSON Data Type: String	01-APP 02-BRW 03-3RI	01-PA 02-NPA	AReq = C	01-PA: Required unless market or regional mandate restricts sending this information. 02-NPA: Required (if available) unless market or regional mandate restricts sending this information.

Data Element/ Field Name	Description	Source	Length/Format/Values	Device Channel	Message Category	Message Inclusion	Conditional Inclusion
Cardholder Billing Address State Field Name: billAddrState	The state or province of the Cardholder billing address associated with the card used for this purchase.	3DS Server	<p>Length: Variable, maximum 3 characters</p> <p>JSON Data Type: String</p> <p>Value accepted:</p> <ul style="list-style-type: none"> • Country subdivision code defined in ISO 3166-2. <p>For example, using the ISO entry US-CA (California, United States), the correct value for this field = CA. Note that the country and hyphen are not included in this value.</p>	01-APP 02-BRW 03-3RI	01-PA 02-NPA	AReq = C	<p>01-PA: Required unless market or regional mandate restricts sending this information, or State is not applicable for this country.</p> <p>02-NPA: Required (if available) unless market or regional mandate restricts sending this information, or State is not applicable for this country.</p>
Cardholder Email Address Field Name: email	The email address associated with the account that is either entered by the Cardholder or is on file with the 3DS Requestor.	3DS Server	<p>Length: Variable, maximum 254 characters</p> <p>JSON Data Type: String</p> <p>Value accepted:</p> <ul style="list-style-type: none"> • Shall meet requirements of Section 3.4 of IETF RFC 5322. 	01-APP 02-BRW 03-3RI	01-PA 02-NPA	AReq = C	Required (if available) unless market or regional mandate restricts sending this information.

Data Element/ Field Name	Description	Source	Length/Format/Values	Device Channel	Message Category	Message Inclusion	Conditional Inclusion
Cardholder Home Phone Number Field Name: homePhone	The home phone number provided by the Cardholder.	3DS Server	<p>Length: Variable</p> <ul style="list-style-type: none"> • cc: 1–3 characters • subscriber: variable, maximum 15 characters <p>Format: JSON Object; strings</p> <p>Values accepted:</p> <ul style="list-style-type: none"> • Country Code and Subscriber sections of the number represented by the following named fields: <ul style="list-style-type: none"> ◦ cc ◦ subscriber <p>Refer to ITU-E.164 for additional information on format and length.</p> <p>Example:</p> <pre>"homePhone": { "cc": "1" , "subscriber": "1234567899" }</pre>	01-APP 02-BRW 03-3RI	01-PA 02-NPA	AReq = C	Required (if available) unless market or regional mandate restricts sending this information.

Data Element/ Field Name	Description	Source	Length/Format/Values	Device Channel	Message Category	Message Inclusion	Conditional Inclusion
Cardholder Information Text Field Name: cardholderInfo	<p>Text provided by the ACS/Issuer to Cardholder during a Frictionless or Decoupled transaction. The Issuer can provide information to Cardholder. For example, “Additional authentication is needed for this transaction, please contact (Issuer Name) at xxx-xxx-xxxx” with optionally the Issuer and Payment System images.</p> <p>Refer to A.20 for UI example.</p>	ACS	<p>Length: Variable JSON Data Type: Object Required:</p> <ul style="list-style-type: none"> • text <ul style="list-style-type: none"> ◦ JSON Data Type: String ◦ Variable, 1–128 characters <p>Optional:</p> <ul style="list-style-type: none"> • issuerImage <ul style="list-style-type: none"> ◦ JSON Data Type: String ◦ Variable, maximum 256 characters ◦ Value accepted: Fully Qualified URL • paymentSystemImage <ul style="list-style-type: none"> ◦ JSON Data Type: String ◦ Variable, maximum 256 characters ◦ Value accepted: Fully Qualified URL 	<p>01-APP 02-BRW 03-3RI</p>	<p>01-PA 02-NPA</p>	<p>ARes = C RReq = O</p>	<p>Required if ACS Decoupled Confirmation Indicator = Y Otherwise, Optional for the ACS.</p>

Data Element/ Field Name	Description	Source	Length/Format/Values	Device Channel	Message Category	Message Inclusion	Conditional Inclusion
Cardholder Mobile Phone Number Field Name: mobilePhone	The mobile phone number provided by the Cardholder.	3DS Server	<p>Length: Variable</p> <ul style="list-style-type: none"> • cc: 1–3 characters • subscriber: variable, maximum 15 characters <p>Format: JSON Object; strings</p> <p>Values accepted:</p> <ul style="list-style-type: none"> • Country Code and Subscriber sections of the number represented by the following named fields: <ul style="list-style-type: none"> ◦ cc ◦ subscriber <p>Refer to ITU-E.164 for additional information on format and length.</p> <p>Example:</p> <pre>"mobilePhone": { "cc": "1" , "subscriber": "123456 7899" }</pre>	01-APP 02-BRW 03-3RI	01-PA 02-NPA	AReq = C	Required (if available) unless market or regional mandate restricts sending this information.

Data Element/ Field Name	Description	Source	Length/Format/Values	Device Channel	Message Category	Message Inclusion	Conditional Inclusion
Cardholder Name Field Name: cardholderName	Name of the Cardholder.	3DS Server	Length: Variable, 1–45 characters JSON Data Type: String	01-APP 02-BRW 03-3RI	01-PA 02-NPA	AReq = C	Required unless market or regional mandate restricts sending this information.
Cardholder Shipping Address City Field Name: shipAddrCity	City portion of the shipping address requested by the Cardholder.	3DS Server	Length: Variable, maximum 50 characters JSON Data Type: String	01-APP 02-BRW 03-3RI	01-PA 02-NPA	AReq = C	Required (if available) unless market or regional mandate restricts sending this information.
Cardholder Shipping Address Country Field Name: shipAddrCountry	Country of the shipping address requested by the Cardholder.	3DS Server	Length: 3 characters JSON Data Type: String Values accepted: <ul style="list-style-type: none">• ISO 3166-1 three-digit country code, other than exceptions listed in Table A.5.	01-APP 02-BRW 03-3RI	01-PA 02-NPA	AReq = C	Required if Cardholder Shipping Address State is present. Required (if available) unless market or regional mandate restricts sending this information.
Cardholder Shipping Address Line 1 Field Name: shipAddrLine1	First line of the street address or equivalent local portion of the shipping address requested by the Cardholder.	3DS Server	Length: Variable, maximum 50 characters JSON Data Type: String	01-APP 02-BRW 03-3RI	01-PA 02-NPA	AReq = C	Required (if available) unless market or regional mandate restricts sending this information.

Data Element/ Field Name	Description	Source	Length/Format/Values	Device Channel	Message Category	Message Inclusion	Conditional Inclusion
Cardholder Shipping Address Line 2 Field Name: shipAddrLine2	The second line of the street address or equivalent local portion of the shipping address requested by the Cardholder.	3DS Server	Length: Variable, maximum 50 characters JSON Data Type: String	01-APP 02-BRW 03-3RI	01-PA 02-NPA	AReq = C	Required (if available) unless market or regional mandate restricts sending this information.
Cardholder Shipping Address Line 3 Field Name: shipAddrLine3	The third line of the street address or equivalent local portion of the shipping address requested by the Cardholder.	3DS Server	Length: Variable, maximum 50 characters JSON Data Type: String	01-APP 02-BRW 03-3RI	01-PA 02-NPA	AReq = C	Required (if available) unless market or regional mandate restricts sending this information.
Cardholder Shipping Address Postal Code Field Name: shipAddrPostCode	The ZIP or other postal code of the shipping address requested by the Cardholder.	3DS Server	Length: Variable, maximum 16 characters JSON Data Type: String	01-APP 02-BRW 03-3RI	01-PA 02-NPA	AReq = C	Required (if available) unless market or regional mandate restricts sending this information.

Data Element/ Field Name	Description	Source	Length/Format/Values	Device Channel	Message Category	Message Inclusion	Conditional Inclusion
Cardholder Shipping Address State Field Name: shipAddrState	The state or province of the shipping address associated with the card being used for this purchase.	3DS Server	<p>Length: Variable: maximum 3 characters</p> <p>JSON Data Type: String</p> <p>Value accepted:</p> <ul style="list-style-type: none"> • Country subdivision code defined in ISO 3166-2. <p>For example, using the ISO entry US-CA (California, United States), the correct value for this field = CA. Note that the country and hyphen are not included in this value.</p>	01-APP 02-BRW 03-3RI	01-PA 02-NPA	AReq = C	Required (if available) unless market or regional mandate restricts sending this information, or State is not applicable for this country.

Data Element/ Field Name	Description	Source	Length/Format/Values	Device Channel	Message Category	Message Inclusion	Conditional Inclusion
Cardholder Work Phone Number Field Name: workPhone	The work phone number provided by the Cardholder.	3DS Server	<p>Length: Variable</p> <ul style="list-style-type: none"> • cc: 1–3 characters • subscriber: variable, maximum 15 characters <p>JSON Data Type: String</p> <p>Values accepted:</p> <ul style="list-style-type: none"> • Country Code and Subscriber sections of the number represented by the following named fields: <ul style="list-style-type: none"> ◦ cc ◦ subscriber <p>Refer to ITU-E.164 for additional information on format and length.</p> <p>Example:</p> <pre>"workPhone": { "cc": "1", "subscriber": "12345 67899" }</pre>	01-APP 02-BRW 03-3RI	01-PA 02-NPA	AReq = C	Required (if available) unless market or regional mandate restricts sending this information.

Data Element/ Field Name	Description	Source	Length/Format/Values	Device Channel	Message Category	Message Inclusion	Conditional Inclusion
Challenge Additional Code Field Name: challengeAddCode	Indicates to the ACS that the Cardholder selected the additional choice.	3DS SDK	Length: 1 character JSON Data Type: String Values accepted: <ul style="list-style-type: none">• Y = Additional choice selected• N = Additional choice not selected	01-APP	01-PA 02-NPA	CReq = C	Required for Native UI if the ACS offers the additional choice button.
Challenge Additional Label Field Name: challengeAddLabel	UI label for the additional choice button provided by the ACS.	ACS	Length: Variable maximum 45 characters JSON Data Type: String	01-APP	01-PA 02-NPA	CRes = C	See Table A.20 for presence conditions.

Challenge Cancelation Indicator Field Name: challengeCancel	Indicator informing the ACS and the DS that the authentication has been cancelled.	3DS SDK ACS	Length: 2 characters JSON Data Type: String Values accepted: <ul style="list-style-type: none"> • 01 = Cardholder selected “Cancel” • 02 = Reserved for future EMVCo use (values invalid until defined by EMVCo). • 03 = Transaction Timed Out—Decoupled Authentication • 04 = Transaction Timed Out at ACS—other timeouts • 05 = Transaction Timed Out at ACS—First CReq not received by ACS • 06 = Transaction Error • 07 = Unknown • 08 = Transaction Timed Out at 3DS SDK • 09 = Error message in response to the CRes message sent by the ACS • 10 = Error message in response to the CReq message received by the ACS 	01-APP 02-BRW 03-3RI	01-PA 02-NPA	CReq = C RReq = C	<ul style="list-style-type: none"> • Required in CReq for 01-APP if the authentication transaction was cancelled by user interaction with the cancellation button in the UI or for other reasons as indicated. • Required in the RReq if the ACS identifies that the authentication transaction was cancelled for reasons as indicated. <p>Value of 04 or 05 is required when Transaction Status Reason = 14.</p>
---	--	--------------------	---	------------------------------------	---------------------	--------------------------	---

Data Element/ Field Name	Description	Source	Length/Format/Values	Device Channel	Message Category	Message Inclusion	Conditional Inclusion
			<ul style="list-style-type: none"> • 11-79 = Reserved for future EMVCo use (values invalid until defined by EMVCo) • 80–99 = Reserved for future DS use 				
Challenge Completion Indicator Field Name: challengeCompletionInd	<p>Indicator of the state of the ACS challenge cycle and whether the challenge has completed or will require additional messages.</p> <p>Shall be populated in all CRes messages to convey the current state of the transaction.</p> <p>Note: If set to Y, the ACS will populate the Transaction Status in the CRes message.</p>	ACS	<p>Length: 1 character</p> <p>JSON Data Type: String</p> <p>Values accepted:</p> <ul style="list-style-type: none"> • Y = Challenge completed, and no further challenge message exchanges are required • N = Challenge not completed and additional challenge message exchanges are required 	01-APP	01-PA 02-NPA	CRes = R	

Data Element/ Field Name	Description	Source	Length/Format/Values	Device Channel	Message Category	Message Inclusion	Conditional Inclusion
Challenge Data Entry Field Name: challengeDataEntry	<p>Contains the data that the Cardholder entered in the Native UI text field.</p> <p>Note: ACS UI Type = 04, 05, 06 and 07 are not supported.</p> <p>Example:</p> <pre>"challengeSelectInfo": [{"phone":"Mobile **** * 321"}, {"mail":"Email a*****g**@g***.co m"}]</pre> <p>The Cardholder selects the phone option:</p> <pre>"challengeDataEntry" :"phone"</pre>	3DS SDK	<p>Length: Variable, maximum 45 characters</p> <p>JSON Data Type: String</p>	01-APP	01-PA 02-NPA	CReq = C	<p>Required when:</p> <ul style="list-style-type: none"> • ACS UI Type = 01, 02, or 03; AND • Challenge data has been entered in the Native UI text field; AND • Challenge Cancelation Indicator is not present; AND • Resend Challenge Information Code is not present; AND • Challenge Additional Code is not present <p>See Table A.16 for Challenge Data Entry conditions.</p>

Data Element/ Field Name	Description	Source	Length/Format/Values	Device Channel	Message Category	Message Inclusion	Conditional Inclusion
Challenge Data Entry 2 Field Name: challengeDataEntryTwo	<p>Contains the data that the Cardholder entered in the Native UI text field.</p> <p>Note: Supported only for ACS UI Type = 01.</p>	3DS SDK	<p>Length: Variable, maximum 45 characters</p> <p>JSON Data Type: String</p>	01-APP	01-PA 02-NPA	CReq = C	<p>Required when:</p> <ul style="list-style-type: none"> • ACS UI Type = 01; AND • Challenge Entry Box 2 object is provided by the ACS; AND • Challenge data has been entered in the second entry box/UI; AND • Challenge Cancelation Indicator is not present; AND • Resend Challenge Information Code is not present; AND • Challenge Additional Code is not present <p>See Table A.16 for Challenge Data Entry conditions.</p>

Data Element/ Field Name	Description	Source	Length/Format/Values	Device Channel	Message Category	Message Inclusion	Conditional Inclusion
Challenge Entry Box Field Name: challengeEntryBox	<p>Defines the setting of an entry box in the Native UI OTP/Text Template:</p> <ul style="list-style-type: none"> • Challenge Data Entry Keyboard Type • Challenge Data Entry Autofill • Challenge Data Entry Autofill Type • Challenge Data Entry Length Maximum • Challenge Data Entry Label • Challenge Data Entry Masking • Challenge Data Entry Masking Toggle 	ACS	<p>Length: Variable JSON Data Type: Object Values accepted:</p> <ul style="list-style-type: none"> • Refer to Table A.26 	01-APP	01-PA 02-NPA	CRes = C	See Table A.20 for conditions.

Data Element/ Field Name	Description	Source	Length/Format/Values	Device Channel	Message Category	Message Inclusion	Conditional Inclusion
Challenge Entry Box 2 Field Name: challengeEntryBoxTwo	<p>Defines the setting of an entry box in the Native UI OTP/Text Template:</p> <ul style="list-style-type: none"> • Challenge Data Entry Keyboard Type • Challenge Data Entry Autofill • Challenge Data Entry Autofill Type • Challenge Data Entry Length Maximum • Challenge Data Entry Label • Challenge Data Entry Masking • Challenge Data Entry Masking Toggle 	ACS	<p>Length: Variable JSON Data Type: Object Values accepted:</p> <ul style="list-style-type: none"> • Refer to Table A.26 	01-APP	01-PA 02-NPA	CRes = C	See Table A.20 for conditions.
Challenge Error Reporting Field Name: challengeErrorReporting	<p>Copy of the Error Message sent or received by the ACS in case of error in the CReq/CRes messages.</p>	ACS	<p>Length: Variable JSON Data Type: Object Values accepted:</p> <ul style="list-style-type: none"> • Refer to Table B.12 for data elements 	01-APP 02-BRW	01-PA 02-NPA	RReq = C	Required when Challenge Cancelation Indicator = 09 or 10.

Data Element/ Field Name	Description	Source	Length/Format/Values	Device Channel	Message Category	Message Inclusion	Conditional Inclusion
Challenge HTML Data Entry Field Name: challengeHTMLDataEntry	Data that the Cardholder entered into the HTML UI. Note: ACS UI Types 01, 02, 03, 04 and 07 are not supported.	3DS SDK	Length: Variable, maximum 256 characters JSON Data Type: String	01-APP	01-PA 02-NPA	CReq = C	Required when: <ul style="list-style-type: none">• ACS UI Type = 05 or 06, AND• Challenge Cancelation Indicator is not present, AND• OOB Continuation Indicator is NOT = 02
Challenge Information Header Field Name: challengeInfoHeader	Header text that for the challenge information screen that is being presented.	ACS	Length: Variable, maximum 45 characters JSON Data Type: String If the field is populated, this information is displayed to the Cardholder.	01-APP	01-PA 02-NPA	CRes = C	See Table A.20 for presence conditions.
Challenge Information Label Field Name: challengeInfoLabel	Text provided to the Cardholder by the ACS/Issuer to specify the expected challenge entry.	ACS	Length: Variable, maximum 45 characters JSON Data Type: String If the field is populated, this information is displayed to the Cardholder.	01-APP	01-PA 02-NPA	CRes = C	See Table A.20 for presence conditions.

Data Element/ Field Name	Description	Source	Length/Format/Values	Device Channel	Message Category	Message Inclusion	Conditional Inclusion
Challenge Information Text Field Name: challengeInfoText	Text provided by the ACS/Issuer to the Cardholder during the Challenge Message exchange.	ACS	<p>Length: Variable, maximum 350 characters</p> <p>JSON Data Type: String</p> <p>If the field is populated, this information is displayed to the Cardholder.</p> <p>Note: Carriage return is supported in this data element and is represented by an "\n".</p> <p>Note: Bold text is supported in this data element and is enclosed between **.</p> <p>Example:</p> <ul style="list-style-type: none"> “This is **bold** text” is rendered as “This is bold text”. 	01-APP	01-PA 02-NPA	CRes = C	See Table A.20 for presence conditions.

Data Element/ Field Name	Description	Source	Length/Format/Values	Device Channel	Message Category	Message Inclusion	Conditional Inclusion
Challenge Information Text Indicator Field Name: challengeInfoTextIndicator	Indicates when the Issuer/ACS would like a warning icon or similar visual indicator to draw attention to the “Challenge Information Text” that is being displayed.	ACS	Length: 1 JSON Data Type: String Value accepted: <ul style="list-style-type: none">• Y = Display indicator• N = Do not display indicator Note: If the field is populated, this information is displayed to the Cardholder.	01-APP	01-PA 02-NPA	CRes = C	See Table A.20 for presence conditions.

Data Element/ Field Name	Description	Source	Length/Format/Values	Device Channel	Message Category	Message Inclusion	Conditional Inclusion
Challenge No Entry Field Name: challengeNoEntry	<p>Indicator informing that the Cardholder submits an empty response (no data entered in the UI).</p> <p>Note: If present this field contains the value Y.</p>	3DS SDK	<p>Length = 1 character JSON Data Type = String Value accepted:</p> <ul style="list-style-type: none"> • Y = No Data Entry 	01-APP	01-PA 02-NPA	CReq = C	<p>Required when:</p> <ul style="list-style-type: none"> • ACS UI Type = 01, 02, or 03, AND • Challenge Data Entry is not present AND • Challenge Data Entry 2 is not present when Challenge Entry Box 2 object was provided by the ACS, AND • Challenge Cancelation Indicator is not present AND • Resend Challenge Information Code is not present AND • Challenge Additional Code is not present

Data Element/ Field Name	Description	Source	Length/Format/Values	Device Channel	Message Category	Message Inclusion	Conditional Inclusion
Challenge Selection Information Field Name: challengeSelectInfo	<p>Selection information that will be presented to the Cardholder if the option is single- or multi-select. The variables will be sent in a JSON Array and parsed by the 3DS SDK for display in the user interface.</p> <p>Example:</p> <pre>"challengeSelectInfo": [{ "phone": "Mobile **** * 321" }, { "mail": "Email a*****g**@g***.co m" }]</pre>	ACS	<p>Size: Variable, 1–8 elements</p> <p>JSON Data Type: Array of Objects.</p> <p>Object: String key/value pair</p> <p>Length: Variable, maximum 45 characters</p> <p>Note: If the field is populated, this information is displayed to the Cardholder.</p>	01-APP	01-PA 02-NPA	CRes = C	See Table A.20 for presence conditions.

Data Element/ Field Name	Description	Source	Length/Format/Values	Device Channel	Message Category	Message Inclusion	Conditional Inclusion
Challenge Window Size Field Name: challengeWindowSize	Dimensions of the challenge iframe that has been displayed to the Cardholder. The ACS shall reply with content that is formatted to appropriately render in this iframe to provide the best possible user experience. Preconfigured sizes are width x height in pixels of the challenge iframe displayed in the Cardholder Browser window.	3DS Requestor	Length: 2 characters JSON Data Type: String Values accepted: <ul style="list-style-type: none">• 01 = 250 x 400• 02 = 390 x 400• 03 = 500 x 600• 04 = 600 x 400• 05 = Full screen	02-BRW	01-PA 02-NPA	CReq = R	

Data Element/ Field Name	Description	Source	Length/Format/Values	Device Channel	Message Category	Message Inclusion	Conditional Inclusion
Default-SDK Type Field Name: defaultSdkType	<p>Indicates the characteristics of a Default-SDK.</p> <p>SDK Variant: SDK implementation characteristics</p> <p>Wrapped Indicator: If the Default-SDK is embedded as a wrapped component in the 3DS Requestor App</p> <p>Example:</p> <pre>"defaultSdkType": { "sdkVariant": "01", "wrappedInd": "Y" }</pre>	3DS Server	<p>JSON Data Type: Object sdkVariant</p> <p>Length: 2 characters</p> <p>JSON Data Type: String</p> <p>Values accepted:</p> <ul style="list-style-type: none"> • 01 = Native • 02–79 = Reserved for EMVCo future use (values invalid until defined by EMVCo) • 80–99 = Reserved for DS use <p>wrappedInd</p> <p>Length: 1 character</p> <p>JSON Data Type: String</p> <p>Value accepted:</p> <ul style="list-style-type: none"> • Y = Wrapped <p>Only present if value = Y</p>	01-APP	01-PA 02-NPA	AReq = C	Required if SDK Type = 01

Data Element/ Field Name	Description	Source	Length/Format/Values	Device Channel	Message Category	Message Inclusion	Conditional Inclusion
Device Binding Data Entry Field Name: deviceBindingDataEntry	Indicator provided by the 3DS SDK to the ACS to confirm whether the Cardholder gives consent to bind the device.	3DS SDK	Length: 1 character JSON Data Type: String Values accepted: <ul style="list-style-type: none">• Y = Consent given to bind device• N = Consent not given to bind device	01-APP	01-PA 02- NPA	CReq = C	Required if Device Binding Information Text was present in the previous CRes message.
Device Binding Information Text Field Name: deviceBindingInfoText	Text provided by the ACS to the Cardholder during the Device Binding process. Example: <ul style="list-style-type: none">• “Would you like to be remembered on this device?”	ACS	Length: Variable, maximum 64 characters	01-APP	01-PA 02-NPA	CRes = C	See Table A.20 for presence conditions.

Device Binding Status Field Name: deviceBindingStatus	Enables the communication of Device Binding Status between the ACS, the DS and the 3DS Requestor. For bound devices (value = 11–14), Device Binding Status also conveys the type of binding that was performed.	3DS Server DS ACS	Length: 2 characters JSON Data Type: String Values accepted: <ul style="list-style-type: none">• 01 = Device is not bound by Cardholder• 02 = Not eligible as determined by issuer• 03 = Pending confirmation by Cardholder• 04 = Cardholder rejected• 05 = Device Binding Status unknown, unavailable, or does not apply• 06 -10 = Reserved for EMVCo future use (values invalid until defined by EMVCo)• 11 = Device is bound by Cardholder (device is bound using hardware / SIM internal to the consumer device. For instance, keys stored in a secure element on the device)• 12 = Device is bound by Cardholder (device is bound using hardware external to	01-APP 02-BRW 03-3RI	01-PA 02-NPA	AReq = O ARes = O RReq = O	
---	--	---------------------------------	--	------------------------------------	---------------------	--	--

Data Element/ Field Name	Description	Source	Length/Format/Values	Device Channel	Message Category	Message Inclusion	Conditional Inclusion
			<p>the consumers device. For example, an external FIDO authenticator</p> <ul style="list-style-type: none">• 13 = Device is bound by Cardholder (Device is bound using data that includes dynamically generated data and could include a unique device ID)• 14 = Device is bound by Cardholder (Device is bound using static device data that has been obtained from the consumers device)• 15 = Device is bound by Cardholder (Other method)• 16–79 = Reserved for EMVCo future use (values invalid until defined by EMVCo)• 80–99 = Reserved for DS use				

Data Element/ Field Name	Description	Source	Length/Format/Values	Device Channel	Message Category	Message Inclusion	Conditional Inclusion
Device Binding Status Source Field Name: deviceBindingStatusSo urce	This data element will be populated by the system setting Device Binding Status.	3DS Server DS ACS	Length: 2 characters JSON Data Type: String Values accepted: <ul style="list-style-type: none">• 01 = 3DS Server• 02 = DS• 03 = ACS• 04-79 = Reserved for EMVCo future use (values invalid until defined by EMVCo)• 80-99 = Reserved for DS use	01-APP 02-BRW 03-3RI	01-PA 02-NPA	AReq = C ARes = C RReq = C	Required if Device Binding Status is present.
Device Channel Field Name: deviceChannel	Indicates the type of channel interface being used to initiate the transaction.	3DS Server	Length: 2 characters JSON Data Type: String Values accepted: <ul style="list-style-type: none">• 01 = App-based (APP)• 02 = Browser (BRW)• 03 = 3DS Requestor Initiated (3RI)• 04–79 = Reserved for EMVCo future use (values invalid until defined by EMVCo)• 80–99 = Reserved for DS use	01-APP 02-BRW 03-3RI	01-PA 02-NPA	AReq = R	

Data Element/ Field Name	Description	Source	Length/Format/Values	Device Channel	Message Category	Message Inclusion	Conditional Inclusion
Device Information Field Name: deviceInfo	<p>Device information gathered by the 3DS SDK from a Consumer Device. This is JSON name/value pairs that as a whole is Base64url encoded.</p> <p>This will be populated by the DS as unencrypted data to the ACS obtained from SDK Encrypted Data.</p>	DS	<p>Length: Variable, maximum 64000 characters</p> <p>JSON Data Type: String</p> <p>Value accepted:</p> <ul style="list-style-type: none"> • Base64url-encoded JSON Object (represented as a string) <p>Refer to <i>EMV 3-D Secure SDK—Device Information</i> for values.</p>	01-APP	01-PA 02-NPA	AReq = C	Required between the DS and ACS but will not be present from 3DS Server to DS.
Device Information Recognised Version Field Name: deviceInfoRecognisedVersion	Indicates the highest Data Version of the Device Information supported by the ACS.	ACS	<p>Length: Variable, minimum 3 characters</p> <p>JSON Data Type: String</p> <p>Values accepted:</p> <ul style="list-style-type: none"> • Any active Device Information Data Version is considered a valid value. <p>Refer to <i>EMV Specification Bulletin 255</i> for values.</p>	01-APP	01-PA 02-NPA	ARes = R	

Data Element/ Field Name	Description	Source	Length/Format/Values	Device Channel	Message Category	Message Inclusion	Conditional Inclusion
Device Rendering Options Supported Field Name: deviceRenderOptions	<p>Identifies the SDK Interface and SDK UI Type that the device supports for displaying specific challenge user interfaces within the 3DS SDK.</p> <p>Note: As established in [Req 314], all Device Rendering Options must be supported by the 3DS SDK.</p>	3DS Server	<p>Length: Variable JSON Data Type: Object Refer to Table A.15 for data elements to include. Note: Data will be formatted into a JSON object prior to being placed into the Device Rendering Options Supported field of the message.</p>	01-APP	01-PA 02-NPA	AReq = R	
DS Protocol Versions Field Name: dsProtocolVersions	<p>Contains the list of active protocol versions supported by the DS.</p> <p>Note: Optional within the Card Range Data (as defined in Table A.6).</p>	DS	<p>Size: Variable, 1–10 elements JSON Data Type: Array of String. String: 5–8 characters Values accepted:</p> <ul style="list-style-type: none"> • Refer to <i>EMV Specification Bulletin 255</i>. 	N/A	N/A	PRes = R	
DS Reference Number Field Name: dsReferenceNumber	EMVCo-assigned unique identifier to track approved DS.	DS	<p>Length: Variable, maximum 32 characters JSON Data Type: String</p>	01-APP 02-BRW 03-3RI	01-PA 02-NPA	<p>AReq = C ARes = R OReq = R</p>	The DS will populate the AReq message with this data element prior to passing to the ACS.

Data Element/ Field Name	Description	Source	Length/Format/Values	Device Channel	Message Category	Message Inclusion	Conditional Inclusion
DS Transaction ID Field Name: dsTransID	Universally unique transaction identifier assigned by the DS to identify a single transaction.	DS	Length: 36 characters JSON Data Type: String Value accepted: <ul style="list-style-type: none"> Canonical format as defined in IETF RFC 4122. May utilise any of the specified versions as long as the output meets specified requirements. 	01-APP 02-BRW 03-3RI	01-PA 02-NPA	AReq = C ARes = R OReq = R ORes = R RReq = R RRes = R PRes = R Erro = C	The DS will populate the AReq with this data element prior to passing to the ACS. Required in Error Message if available (e.g., can be obtained from a message or is being generated).
DS URL Field Name: dsURL	URL of the DS to which the ACS will send the RReq if a challenge occurs. The ACS is responsible for storing this value for later use in the transaction for sending the RReq to the DS.	DS	Length: Variable, maximum 2048 characters JSON Data Type: String Value accepted: <ul style="list-style-type: none"> Fully Qualified URL Example: <ul style="list-style-type: none"> https://server.domainname.com 	01-APP 02-BRW 03-3RI	01-PA 02-NPA	AReq = C	Required between the DS and ACS but will not be present from 3DS Server to DS.

Data Element/ Field Name	Description	Source	Length/Format/Values	Device Channel	Message Category	Message Inclusion	Conditional Inclusion
DS URL List Field Name: dsUrlList	List of DS URLs to which the 3DS Server will send the AReq message. The DS optionally provides this list in case there are preferred DS URLs for some countries.	DS	Size: Variable, 1–10 elements JSON Data Type: Array of Objects Values accepted: <ul style="list-style-type: none">• See Table A.7 for DS URL List.	N/A	N/A	PRes = O	
Electronic Commerce Indicator (ECI) Field Name: eci	Payment System-specific value provided by the ACS or DS to indicate the results of the attempt to authenticate the Cardholder.	ACS DS	Length: 2 characters JSON Data Type: String Values accepted: <ul style="list-style-type: none">• Payment System-specific	01-APP 02-BRW 03-3RI	01-PA 02-NPA	ARes = C RReq = C	The requirements for the presence of this field are DS specific.
Error Code Field Name: errorCode	Code indicating the type of problem identified in the message.		Length: 3 characters JSON Data Type: String Values accepted: <ul style="list-style-type: none">• See Table A.4 for values.	N/A	N/A	Erro = R	
Error Component Field Name: errorComponent	Code indicating the 3-D Secure component that identified the error.		Length: 1 character JSON Data Type: String Values accepted: <ul style="list-style-type: none">• C = 3DS SDK• S = 3DS Server• D = DS• A = ACS	N/A	N/A	Erro = R	

Data Element/ Field Name	Description	Source	Length/Format/Values	Device Channel	Message Category	Message Inclusion	Conditional Inclusion
Error Description Field Name: errorDescription	Text describing the problem identified in the message.		Length: Variable, maximum 2048 characters JSON Data Type: String	N/A	N/A	Erro = R	
Error Detail Field Name: errorDetail	Additional detail regarding the problem identified in the message.		Length: Variable, maximum 2048 characters JSON Data Type: String	N/A	N/A	Erro = R	
Error Message Type Field Name: errorMessageType	Identifies the Message Type that was identified as erroneous.		Length: 4 characters JSON Data Type: String Values accepted: <ul style="list-style-type: none">• See Message Type	N/A	N/A	Erro = C	Conditional on Message Type being recognisable.
EMV Payment Token Indicator Field Name: payTokenInd	A value of True indicates that the transaction was de-tokenised prior to being received by the ACS. This data element will be populated by the system residing in the 3-D Secure domain where the de-tokenisation occurs (i.e., the 3DS Server or the DS). Note: The Boolean value of true is the only valid response for this field when it is present.	3DS Server DS	JSON Data Type: Boolean Value accepted: <ul style="list-style-type: none">• true	01-APP 02-BRW 03-3RI	01-PA 02-NPA	AReq = C	Required if there is a de-tokenisation of an Account Number.

Data Element/ Field Name	Description	Source	Length/Format/Values	Device Channel	Message Category	Message Inclusion	Conditional Inclusion
EMV Payment Token Information Field Name: payTokenInfo	Information about de-tokenised Payment Token.	3DS Server DS	Length: Variable JSON Data Type: Object Refer to Table A.25 for data elements to include. Note: Data will be formatted into a JSON object prior to being placed into the EMV Payment Token field of the message.	01-APP 02-BRW 03-3RI	01-PA 02-NPA	AReq = O	
EMV Payment Token Source Field Name: payTokenSource	This data element will be populated by the system residing in the 3-D Secure domain where the de-tokenisation occurs.	3DS Server DS	Length: 2 characters JSON Data Type: String Values accepted: <ul style="list-style-type: none">• 01 = 3DS Server• 02 = DS• 03-79 = Reserved for EMVCo future use (values invalid until defined by EMVCo)• 80-99 = Reserved for DS use	01-APP 02-BRW 03-3RI	01-PA 02-NPA	AReq = C	Required if EMV Payment Token Indicator = true.
Expandable Information Label Field Name: expandInfoLabel	Label displayed to the Cardholder for the content in Expandable Information Text.	ACS	Length: Variable, maximum 45 characters JSON Data Type: String	01-APP	01-PA 02-NPA	CRes = O	

Data Element/ Field Name	Description	Source	Length/Format/Values	Device Channel	Message Category	Message Inclusion	Conditional Inclusion
Expandable Information Text Field Name: expandInfoText	Text provided by the Issuer from the ACS to be displayed to the Cardholder for additional information and the format will be an expandable text field.	ACS	Length: Variable, maximum 256 characters JSON Data Type: String Note: Carriage return is supported in this data element and is represented by an “\n”. Note: Bold text is supported in this data element and is enclosed between **. For example “This is **bold** text” is rendered as This is bold text.	01-APP	01-PA 02-NPA	CRes = O	
Information Continuation Indicator Field Name: infoContinueIndicator	Indicator notifying the ACS that the Cardholder selected the Information Continue button in the Information UI template. Note: The Boolean value of true is the only valid response for this field when it is present.	3DS SDK	JSON Data Type: Boolean Value accepted: <ul style="list-style-type: none">• true	01-APP	01-PA 02-NPA	CReq = C	Required for ACS UI Type = 07 if the Cardholder selects the button on the device.
Information Continuation Label Field Name: infoContinueLabel	UI label used for the button that the Cardholder selects in the Information UI template.	ACS	Length: Variable, maximum 45 characters JSON Data Type: String	01-APP	01-PA 02-NPA	CRes = C	See Table A.20 for presence conditions.

Data Element/ Field Name	Description	Source	Length/Format/Values	Device Channel	Message Category	Message Inclusion	Conditional Inclusion
Instalment Payment Data Field Name: purchaseInstalData	Indicates the maximum number of authorisations permitted for instalment payments.	3DS Server	Length: Variable, maximum 3 characters JSON Data Type: String Values accepted: <ul style="list-style-type: none">• Value shall be greater than 1 Example values accepted: <ul style="list-style-type: none">• 2• 02• 002	01-APP 02-BRW 03-3RI	01-PA 02-NPA	AReq = C	<ul style="list-style-type: none">• Required if Merchant and Cardholder have agreed to instalment payments, i.e., if 3DS Requestor Authentication Indicator = 03. Omitted if not an instalment payment authentication• Required for 03-3RI if 3RI Indicator = 02.
Interaction Counter Field Name: interactionCounter	Indicates the number of authentication cycles (excluding Decoupled Authentication) attempted by the Cardholder. Value to be tracked by the ACS.	ACS	Length: 2 characters JSON Data Type: String	01-APP 02-BRW	01-PA 02-NPA	RReq = C	Required unless ACS Decoupled Confirmation Indicator = Y.
Issuer Image Field name: issuerImage	Sent in the initial CRes message from the ACS to the 3DS SDK to provide the URL(s) of the Issuer logo or image to be used in the Native UI.	ACS	Format: JSON Object Values accepted: <ul style="list-style-type: none">• Refer to Table A.21 for data elements.	01-APP	01-PA 02-NPA	CRes = C	Presence of this field is Payment System specific. Absent for ACS UI Type = 05 and 06.

Data Element/ Field Name	Description	Source	Length/Format/Values	Device Channel	Message Category	Message Inclusion	Conditional Inclusion
Merchant Category Code Field Name: <code>mcc</code>	DS-specific code describing the Merchant's type of business, product or service.	3DS Server	Length: 4 characters JSON Data Type: String This value correlates to the Merchant Category Code as defined by each Payment System or DS.	01-APP 02-BRW 03-3RI	01-PA 02-NPA	01-PA: AReq = R 02-NPA: AReq = O	Optional but strongly recommended to include for 02-NPA if the merchant is also the 3DS Requestor.
Merchant Country Code Field Name: <code>merchantCountryCode</code>	Country Code of the Merchant. This value correlates to the Merchant Country Code as defined by each Payment System or DS.	3DS Server	Length: 3 characters JSON Data Type: String Values accepted: <ul style="list-style-type: none">• ISO 3166-1 numeric three-digit country code, other than exceptions listed in Table A.5. Note: The same value must be used in the authorisation request.	01-APP 02-BRW 03-3RI	01-PA 02-NPA	01-PA: AReq = R 02-NPA: AReq = O	Optional but strongly recommended to include for 02-NPA if the merchant is also the 3DS Requestor.
Merchant Name Field Name: <code>merchantName</code>	Merchant name assigned by the Acquirer or Payment System.	3DS Server	Length: Variable, maximum 40 characters JSON Data Type: String Same name used in the authorisation message as defined in ISO 8583-1.	01-APP 02-BRW 03-3RI	01-PA 02-NPA	01-PA: AReq = R 02-NPA: AReq = O	Optional but strongly recommended to include for 02-NPA if the merchant is also the 3DS Requestor.

Data Element/ Field Name	Description	Source	Length/Format/Values	Device Channel	Message Category	Message Inclusion	Conditional Inclusion
Merchant Risk Indicator Field Name: merchantRiskIndicator	Merchant's assessment of the level of fraud risk for the specific authentication for both the cardholder and the authentication being conducted.	3DS Server	Length: Variable JSON Data Type: Object Refer to Table A.11 for data elements. Note: Data will be formatted into a JSON object prior to being placed into the Merchant Risk Indicator field of the message.	01-APP 02-BRW 03-3RI	01-PA 02-NPA	AReq = O	Optional, but strongly recommended to include.
Message Category Field Name: messageCategory	Identifies the category of the message for a specific use case.	3DS Server	Length: 2 characters JSON Data Type: String Values accepted: <ul style="list-style-type: none"> • 01 = PA • 02 = NPA • 03–79 = Reserved for EMVCo future use (values invalid until defined by EMVCo) • 80–99 = Reserved for DS use 	01-APP 02-BRW 03-3RI	01-PA 02-NPA	AReq = R RReq = R	

Data Element/ Field Name	Description	Source	Length/Format/Values	Device Channel	Message Category	Message Inclusion	Conditional Inclusion
Message Extension Field Name: messageExtension	Data necessary to support requirements not otherwise defined in the 3-D Secure message are carried in a Message Extension.	3DS Server 3DS SDK ACS DS	Size: Variable, maximum 1–15 elements JSON Data Type: Array of Objects Values accepted: <ul style="list-style-type: none">• Refer to Table A.9 for data elements.	01-APP 02-BRW 03-3RI	01-PA 02-NPA	AReq = C ARes = C CReq = C CRes = C OReq = C ORes = C PReq = C PRes = C RReq = C RRes = C	Conditions to be set by each DS.

Data Element/ Field Name	Description	Source	Length/Format/Values	Device Channel	Message Category	Message Inclusion	Conditional Inclusion
Message Type Field Name: messageType	Identifies the type of message that is passed.	3DS Server 3DS SDK DS ACS	Length: 4 characters JSON Data Type: String Values accepted: <ul style="list-style-type: none">• AReq• ARes• CReq• CRes• OReq• ORes• PReq• PRes• RReq• RRes• Erro	01-APP 02-BRW 03-3RI	01-PA 02-NPA	AReq = R ARes = R CReq = R CRes = R OReq = R ORes = R PReq = R PRes = R RReq = R RRes = R Erro = R	

Data Element/ Field Name	Description	Source	Length/Format/Values	Device Channel	Message Category	Message Inclusion	Conditional Inclusion
Message Version Number Field Name: messageVersion	<p>Protocol version identifier This shall be the Protocol Version Number of the specification utilised by the system creating this message.</p> <p>The Message Version Number is set by the 3DS Server which originates the protocol with the AReq message. The Message Version Number does not change during a 3DS transaction.</p>	3DS Server	<p>Length: Variable, 5–8 characters JSON Data Type: String Value accepted:</p> <ul style="list-style-type: none"> • Major.minor.patch <p>Example:</p> <ul style="list-style-type: none"> • 99.99.99. <p>Refer to <i>EMV Specification Bulletin 255</i>.</p>	01-APP 02-BRW 03-3RI	01-PA 02-NPA	AReq = R ARes = R CReq = R CRes = R OReq = R ORes = R PReq = R PRes = R RReq = R RRes = R Erro = R	
Multi-Transaction Field Name: multiTransaction	Additional transaction information in case of multiple transactions or merchants.	3DS Server	Length: Variable JSON Data Type: Object Refer to Table A.18 for data elements to include.	01-APP 02-BRW 03-3RI	01-PA 02-NPA	AReq = O	
Notification URL Field Name: notificationURL	Fully Qualified URL of the system that receives the CRes message or Error Message. The CRes message is posted by the ACS through the Cardholder Browser at the end of the challenge and receipt of the RRes message.	3DS Server	Length: Variable, maximum 256 characters JSON Data Type: String Value accepted: <ul style="list-style-type: none"> • Fully Qualified URL 	02-BRW	01-PA 02-NPA	AReq = R	

Data Element/ Field Name	Description	Source	Length/Format/Values	Device Channel	Message Category	Message Inclusion	Conditional Inclusion
OOB App Label Field Name: oobAppLabel	<p>Label to be displayed for the link to the OOB App URL.</p> <p>Example:</p> <pre>"oobAppLabel": "Open Your Bank App"</pre>	ACS	<p>Length: Variable, maximum 45 characters</p> <p>JSON Data Type: String</p>	01-APP	01-PA 02-NPA	CRes = C	Required if the OOB App URL is available and ACS UI Type = 04.
OOB App Status Field Name: oobAppStatus	<p>Status code indicating the problem type encountered when using the OOB App URL.</p>	3DS SDK	<p>Length: Variable, maximum 2 characters</p> <p>JSON Data Type: String</p> <p>Values accepted:</p> <ul style="list-style-type: none"> • 01 = Open OOB App URL failed • 02–99 = Reserved for future EMVCo use (values invalid until defined by EMVCo) 	01-APP	01-PA 02-NPA	CReq = C	Required if the Cardholder encountered an error when selecting the OOB App URL for ACS UI Type = 04 or 06.

Data Element/ Field Name	Description	Source	Length/Format/Values	Device Channel	Message Category	Message Inclusion	Conditional Inclusion
OOB App URL Field Name: oobAppURL	<p>Universal App Link to an authentication app used in the OOB authentication. The OOB App URL will open the appropriate location within the OOB Authentication App.</p> <p>Refer to Table 1.3 for the Universal App Link definition.</p>	ACS	<p>Length: Variable, maximum 2048 characters</p> <p>JSON Data Type: String</p> <p>Value accepted:</p> <ul style="list-style-type: none"> • Universal App Link 	01-APP	01-PA 02-NPA	CRes = C	<p>Required for ACS UI Type = 04 or 06 if:</p> <ul style="list-style-type: none"> • OOB App URL Indicator = 01 in the CReq message; AND • the ACS utilises the OOB Authentication App automatic switching feature

Data Element/ Field Name	Description	Source	Length/Format/Values	Device Channel	Message Category	Message Inclusion	Conditional Inclusion
OOB App URL Indicator Field Name: oobAppURLInd	Indicates if the 3DS SDK supports the OOB App URL.	3DS SDK	<p>Length: Variable, maximum 48 characters</p> <p>JSON Data Type: String</p> <p>Values accepted:</p> <ul style="list-style-type: none"> • 01 = Supported • 02 = Not supported by the device • 03 = Not supported by the 3DS Requestor • 04–79 = Reserved for EMVCo future use (values invalid until defined by EMVCo) • 80–99 = Reserved for DS use <p>If SDK Type = 02 and Split-SDK Type = 02, the OOB App URL Indicator is set to 02.</p> <p>Note: OOB App URL does not work for the Split-SDK/Browser.</p>	01-APP	01-PA 02-NPA	CReq = R	

Data Element/ Field Name	Description	Source	Length/Format/Values	Device Channel	Message Category	Message Inclusion	Conditional Inclusion
OOB Continuation Indicator Field Name: oobContinue	Indicator notifying the ACS that the Cardholder has selected the OOB Continuation button in an OOB authentication method, or that the 3DS SDK automatically completes without any Cardholder interaction.	3DS SDK	Length: 2 characters JSON Data Type: String Values accepted: <ul style="list-style-type: none">• 01 = Cardholder clicks the button• 02 = Automatic complete• 03–99 = Reserved for EMVCo future use (values invalid until defined by EMVCo)	01-APP	01-PA 02-NPA	CReq = C	Required if: <ul style="list-style-type: none">• ACS UI Type = 04; OR• ACS UI Type = 06 when the 3DS SDK sends a CReq message unless Challenge Additional Code = Y.
OOB Continuation Label Field Name: oobContinueLabel	Label to be used in the UI for the button that the user selects when they have completed the OOB authentication.	ACS	Length: Variable, maximum 45 characters JSON Data Type: String	01-APP	01-PA 02-NPA	CRes = C	Required for ACS UI Type = 04 if the ACS utilises OOB manual switching.

Data Element/ Field Name	Description	Source	Length/Format/Values	Device Channel	Message Category	Message Inclusion	Conditional Inclusion
Operation Category Field Name: opCategory	Indicates the category/type of information.	DS	Length: 2 characters JSON Data Type: String Values accepted: <ul style="list-style-type: none"> • 01 = General • 02 = Operational alert • 03 = Public Key Certificate expiry • 04 = LOA/AOC expiry • 05 = Fraud • 06 = Other • 07–79 = Reserved for EMVCo future use (values invalid until defined by EMVCo) • 80–99 = Reserved for DS use 	N/A	N/A	OReq = R	
Operation Description Field Name: opDescription	Describes the reason for the operational communication or the response to an action taken by the recipient.	DS	Length: Variable, maximum 20000 characters JSON Data Type: String	N/A	N/A	OReq = R	
Operation Expiration Date Field Name: opExpDate	The date after which the relevance of the operational information (e.g., certificate expiration dates, SLAs, etc.) expires.	DS	Length: 8 characters JSON Data Type: String Format accepted: <ul style="list-style-type: none"> • YYYYMMDD 	N/A	N/A	OReq = R	

Data Element/ Field Name	Description	Source	Length/Format/Values	Device Channel	Message Category	Message Inclusion	Conditional Inclusion
Operation Message Status Field Name: opStatus	Indicates the status of the Operation Request message sequence from the source of the OReq.	3DS Server ACS	Length: 2 characters JSON Data Type: String Values accepted: <ul style="list-style-type: none">• 01 = Successfully received messages• 02 = Message sequence is broken• 03 = Requested action is not supported or not executed by the 3DS Server or ACS when OReq message was received• 04–79 = Reserved for EMVCo future use (values invalid until defined by EMVCo)• 80–99 = Reserved for DS use	N/A	N/A	ORes = R	

Operation Prior Transaction Reference Field Name: opPriorTransRef	This data element provides additional information enabling the recipient to reference a prior transaction.	DS	<p>JSON Data Type: Object</p> <ul style="list-style-type: none">• transIdType: 2 characters<ul style="list-style-type: none">◦ 01 = 3DS Server◦ 02 = DS◦ 03 = ACS• transId: 36 characters<ul style="list-style-type: none">◦ Canonical format as defined in IETF RFC 4122. May utilise any of the specified versions as long as the output meets specified requirements. <p>For example, a prior DS Transaction ID would be represented as:</p> <pre>"opPriorTransRef": [{ "transIdType": "02", "transId": "4317fdc3-ad24-5443-8000-00000000891" }]</pre>	N/A	N/A	OReq = O	
---	--	----	--	-----	-----	----------	--

Data Element/ Field Name	Description	Source	Length/Format/Values	Device Channel	Message Category	Message Inclusion	Conditional Inclusion
Operation Sequence Field Name: opSeq	<p>Indicates the current and total messages in an OReq message sequence.</p> <p>seqId: This element uniquely identifies a message sequence and will remain constant in the sequence of messages.</p> <p>seqNum: This element represents the current message in the sequence.</p> <p>seqTotal: This element represents the total number of messages in the sequence and will remain constant in the sequence of messages.</p>	DS	<p>JSON Data Type: Object</p> <ul style="list-style-type: none"> • seqId: 36 characters <ul style="list-style-type: none"> ◦ Canonical format as defined in IETF RFC 4122. May utilise any of the specified versions as long as the output meets specified requirements. • seqNum: 2 characters Values accepted:<ul style="list-style-type: none"> ◦ 01–99 • seqTotal: 2 characters Values accepted<ul style="list-style-type: none"> ◦ 01–99 <p>For example, the first of three messages in an OReq sequence would be represented as:</p> <pre>"opSeq": { "seqId": "4317fdc3-ad24-5443-8000-00000000891", "seqNum": "01", "seqTotal": "03"}</pre>	N/A	N/A	OReq = R	

Data Element/ Field Name	Description	Source	Length/Format/Values	Device Channel	Message Category	Message Inclusion	Conditional Inclusion
Operation Severity Field Name: opSeverity	<p>Indicates the importance/severity level of the operational information.</p> <p>Critical = Immediate action to be taken by recipient</p> <p>Major = Major impact; Upcoming action to be taken by recipient</p> <p>Minor = Minor impact; Upcoming action to be taken by recipient</p> <p>Informational = Informational only, with no immediate action by recipient</p>	DS	<p>Length: 2 characters</p> <p>JSON Data Type: String</p> <p>Values accepted:</p> <ul style="list-style-type: none"> • 01 = Critical • 02 = Major • 03 = Minor • 04 = Informational • 05–79 = Reserved for EMVCo future use (values invalid until defined by EMVCo) • 80–99 = Reserved for DS use 	N/A	N/A	OReq = R	
Payee Origin Field Name: payeeOrigin	<p>The origin of the payee that will be provided in the SPC Transaction Data Refer to Secure Payment Confirmation.</p>	3DS Server	<p>Length: Variable, maximum 2048 characters</p> <p>JSON Data Type: String</p> <p>Value accepted:</p> <ul style="list-style-type: none"> • Fully Qualified URL 	02-BRW	01-PA 02-NPA	AReq = C	Required if 3DS Requestor SPC Support = Y
Payment System Image Field Name: psImage	Sent in the initial CRes message from the ACS to the 3DS SDK to provide the URL(s) of the DS or Payment System logo or image to be used in the Native UI.	ACS	<p>JSON Data Type: Object</p> <p>Values accepted:</p> <ul style="list-style-type: none"> • Refer to Table A.22 for data elements. 	01-APP	01-PA 02-NPA	CRes = C	<p>Presence of this field are Payment System-specific.</p> <p>Absent for ACS UI Type = 05 and 06.</p>

Data Element/ Field Name	Description	Source	Length/Format/Values	Device Channel	Message Category	Message Inclusion	Conditional Inclusion
Purchase Amount Field Name: purchaseAmount	<p>Purchase amount in minor units of currency with all punctuation removed.</p> <p>When used in conjunction with the Purchase Currency Exponent field, proper punctuation can be calculated.</p>	3DS Server	<p>Length: Variable, maximum 48 characters</p> <p>JSON Data Type: String</p> <p>Example:</p> <p>Purchase amount is USD 123.45</p> <p>Example values accepted:</p> <ul style="list-style-type: none"> • 12345 • 012345 • 0012345 	01-APP 02-BRW 03-3RI	01-PA 02-NPA	01-PA: AReq = R 02-NPA: AReq = C	<ul style="list-style-type: none"> • Required for 02-NPA if 3DS Requestor Authentication Indicator = 02, 03, 07, 08, 09 • Required for 02-NPA if 3RI Indicator = 01, 02, 06, 07, 08, 09, 11, 15
Purchase Currency Field Name: purchaseCurrency	Currency in which purchase amount is expressed.	3DS Server	<p>Length: 3 characters; numeric</p> <p>JSON Data Type: String</p> <p>Values accepted:</p> <ul style="list-style-type: none"> • ISO 4217 three-digit currency codes, other than those listed in Table A.5. 	01-APP 02-BRW 03-3RI	01-PA 02-NPA	01-PA: AReq = R 02-NPA: AReq = C	<ul style="list-style-type: none"> • Required for 02-NPA if 3DS Requestor Authentication Indicator = 02, 03, 07, 08, 09 • Required for 02-NPA if 3RI Indicator = 01, 02, 06, 07, 08, 09, 11, 15

Data Element/ Field Name	Description	Source	Length/Format/Values	Device Channel	Message Category	Message Inclusion	Conditional Inclusion
Purchase Currency Exponent Field Name: purchaseExponent	Minor units of currency as specified in the ISO 4217 currency exponent. Example: <ul style="list-style-type: none">• USD = 2• Yen = 0	3DS Server	Length: 1 character JSON Data Type: String	01-APP 02-BRW 03-3RI	01-PA 02-NPA	01-PA: AReq = R 02-NPA: AReq = C	<ul style="list-style-type: none"> • Required for 02-NPA if 3DS Requestor Authentication Indicator = 02, 03, 07, 08, 09 • Required for 02-NPA if 3RI Indicator = 01, 02, 06, 07, 08, 09, 11, 15
Purchase Date & Time Field Name: purchaseDate	Date and time of the authentication converted into UTC.	3DS Server	Length: 14 characters JSON Data Type: String Format accepted: YYYYMMDDHHMMSS	01-APP 02-BRW 03-3RI	01-PA 02-NPA	01-PA: AReq = R 02-NPA: AReq = C	<ul style="list-style-type: none"> • Required for 02-NPA if 3DS Requestor Authentication Indicator = 02, 03, 07, 08, 09 • Required for 02-NPA if 3RI Indicator = 01, 02, 06, 07, 08, 09, 11, 15

Data Element/ Field Name	Description	Source	Length/Format/Values	Device Channel	Message Category	Message Inclusion	Conditional Inclusion
Read Order Field Name: readOrder	Indicates the order in which to process the card range records from the PRes message.	DS	Length: 2 characters JSON Data Type: String Values accepted: <ul style="list-style-type: none">• 01 = Direct order/FIFO (First In First Out)• 02 = Reverse order/LIFO (Last In First Out)• 03-79 = Reserved for EMVCo future use (values invalid until defined by EMVCo)• 80-99 = Reserved for DS use	N/A	N/A	PRes = R	

Data Element/ Field Name	Description	Source	Length/Format/Values	Device Channel	Message Category	Message Inclusion	Conditional Inclusion
Recurring Amount Field Name: recurringAmount	Recurring amount in minor units of currency with all punctuation removed.	3DS Server	Length: Variable, maximum 48 characters JSON Data Type: String Example: Purchase amount is USD 123.45 Example values accepted: <ul style="list-style-type: none">• 12345• 012345• 0012345	01-APP 02-BRW 03-3RI	01-PA 02-NPA	AReq = C	Required if: <ul style="list-style-type: none">• [3DS Requestor Authentication Indicator = 02 or 03; OR 3RI Indicator = 01 or 02] AND• Recurring Indicator/Amount Indicator = 01
Recurring Currency Field Name: recurringCurrency	Currency in which the Recurring Amount is expressed.	3DS Server	Length: 3 characters; numeric JSON Data Type: String Values accepted: <ul style="list-style-type: none">• ISO 4217 three-digit currency codes, other than those listed in Table A.5.	01-APP 02-BRW 03-3RI	01-PA 02-NPA	AReq = C	Required if the Recurring Amount is present.

Data Element/ Field Name	Description	Source	Length/Format/Values	Device Channel	Message Category	Message Inclusion	Conditional Inclusion
Recurring Currency Exponent Field Name: recurringExponent	Minor units of currency as specified in the ISO 4217 currency exponent. Examples: <ul style="list-style-type: none">• USD = 2• JPY = 0	3DS Server	Length: 1 character; numeric JSON Data Type: String	01-APP 02-BRW 03-3RI	01-PA 02-NPA	AReq = C	Required if the Recurring Amount is present.
Recurring Date Field Name: recurringDate	Effective date of the new authorised amount following the first/promotional payment in a recurring or instalment transaction.	3DS Server	Length: 8 characters JSON Data Type: String Date format accepted: <ul style="list-style-type: none">• YYYYMMDD	01-APP 02-BRW 03-3RI	01-PA 02-NPA	AReq = C	Required if Recurring Indicator/Frequency Indicator = 01.
Recurring Expiry Field Name: recurringExpiry	Date after which no further authorisations are performed.	3DS Server	Length: 8 characters JSON Data Type: String Date format accepted: <ul style="list-style-type: none">• YYYYMMDD	01-APP 02-BRW 03-3RI	01-PA 02-NPA	AReq = C	Required if there is an end date.

Data Element/ Field Name	Description	Source	Length/Format/Values	Device Channel	Message Category	Message Inclusion	Conditional Inclusion
Recurring Frequency Field Name: recurringFrequency	Indicates the minimum number of days between authorisations for a recurring or instalment transaction.	3DS Server	Length: Variable, maximum 4 characters JSON Data Type: String Values accepted: <ul style="list-style-type: none">• Numeric values between 1 and 9999 Example values accepted: <ul style="list-style-type: none">• 31• 031• 0031	01-APP 02-BRW 03-3RI	01-PA 02-NPA	AReq = C	Required if Recurring Indicator/ Frequency Indicator = 01.

Data Element/ Field Name	Description	Source	Length/Format/Values	Device Channel	Message Category	Message Inclusion	Conditional Inclusion
Recurring Indicator Field Name: recurringInd	<p>Indicates whether the recurring or instalment payment has a fixed or variable amount and frequency.</p> <p>The Recurring Indicator object contains:</p> <ul style="list-style-type: none"> the Amount Indicator the Frequency Indicator <p>Example:</p> <pre>{ "recurringInd": { "amountInd": "01", "frequencyInd": "02" }}</pre>	3DS Server	<p>JSON Data Type: Object</p> <p>Amount Indicator Field Name: amountInd Values accepted:</p> <ul style="list-style-type: none"> 01 = Fixed Purchase Amount 02 = Variable Purchase Amount 03–79 = Reserved for EMVCo future use (values invalid until defined by EMVCo) 80–99 = Reserved for DS use <p>Frequency Indicator Field Name: frequencyInd Values accepted:</p> <ul style="list-style-type: none"> 01 = Fixed Frequency 02 = Variable or Unknown Frequency 03–79 = Reserved for EMVCo future use (values invalid until defined by EMVCo) 80–99 = Reserved for DS use 	01-APP 02-BRW 03-3RI	01-PA 02-NPA	AReq = C	<p>Required if:</p> <ul style="list-style-type: none"> 3DS Requestor Authentication Indicator = 02 or 03; OR 3RI Indicator = 01 or 02

Data Element/ Field Name	Description	Source	Length/Format/Values	Device Channel	Message Category	Message Inclusion	Conditional Inclusion
Resend Challenge Information Code Field Name: resendChallenge	Indicator to the ACS that the Cardholder selected the Resend Information button.	3DS SDK	Length: 1 character JSON Data Type: String Value accepted: • Y = Resend	01-APP	01-PA 02-NPA	CReq = C	Required if the Cardholder is requesting the ACS to resend challenge information (value = Y) AND ACS UI Type = 01.
Resend Information Label Field Name: resendInformationLabel	Label to be used in the UI for the button that the user selects when they would like to have the authentication information resent.	ACS	Length: Variable maximum 45 characters JSON Data Type: String	01-APP	01-PA 02-NPA	CRes = C	See Table A.20 for inclusion conditions.

Data Element/ Field Name	Description	Source	Length/Format/Values	Device Channel	Message Category	Message Inclusion	Conditional Inclusion
Results Message Status Field Name: resultsStatus	<p>Indicates the status of the Results Request message from the 3DS Server to provide additional data to the ACS.</p> <p>This will indicate if the message was successfully received for further processing or will be used to provide more detail on why the Challenge could not be completed from the 3DS Client to the ACS.</p>	3DS Server	<p>Length: 2 characters</p> <p>JSON Data Type: String</p> <p>Values accepted:</p> <ul style="list-style-type: none"> • 01 = RReq received for further processing • 02 = CReq not sent to ACS by 3DS Requestor (3DS Server or 3DS Requestor opted out of the challenge) • 03 = ARes (Transaction Status = C or D) not delivered to the 3DS Requestor due to technical error • 04 = 3DS Server will process Decoupled Authentication in a subsequent authentication • 05–79 = Reserved for EMVCo future use (values invalid until defined by EMVCo) • 80–99 = Reserved for DS use 	<p>01-APP 02-BRW 03-3RI</p>	<p>01-PA 02-NPA</p>	RRes = R	

Data Element/ Field Name	Description	Source	Length/Format/Values	Device Channel	Message Category	Message Inclusion	Conditional Inclusion
SDK App ID Field Name: sdkAppID	Universally unique ID created upon all installations of the 3DS Requestor App on a Consumer Device. This will be newly generated and stored by the 3DS SDK for each installation. Note: In case of Browser-SDK, the SDK App ID value is not reliable, and may change for each transaction.	3DS SDK (sent via 3DS Server)	Length: 36 characters JSON Data Type: String Canonical format as defined in IETF RFC 4122. This may utilise any of the specified versions as long as the output meets specified requirements.	01-APP	01-PA 02-NPA	AReq = R	
SDK Counter SDK to ACS Field Name: sdkCounterStoA	Counter used as a security measure in the 3DS SDK to ACS secure channel. Note: The counter is the decimal value equivalent of the byte, encoded as a numeric string.	3DS SDK	Length: 3 characters JSON Data Type: String Values accepted: • 000–255	01-APP	01-PA 02-NPA	CReq = R	
SDK Encrypted Data Field Name: sdkEncData	JWE Object (represented as a string) as defined in Section 6.2.2.1 containing data encrypted by the 3DS SDK for the DS to decrypt.	3DS SDK (sent via 3DS Server)	Length: Variable, maximum 64000 characters JSON Data Type: String	01-APP	01-PA 02-NPA	AReq = C	Required 3DS Server to DS but will not be present DS to ACS.

Data Element/ Field Name	Description	Source	Length/Format/Values	Device Channel	Message Category	Message Inclusion	Conditional Inclusion
SDK Ephemeral Public Key (QC) Field Name: sdkEphemPubKey	<p>Public key component of the ephemeral key pair generated by the 3DS SDK and used to establish session keys between the 3DS SDK and ACS.</p> <p>In AReq, this data element is present as its own object.</p> <p>In ARes, this data element is contained within the ACS Signed Content JWS Object.</p> <p>See Section 6.2.3.1 for additional detail.</p>	3DS SDK	<p>Length: Variable, maximum 256 characters</p> <p>JSON Data Type: Object JWK</p>	01-APP	01-PA 02-NPA	AReq = R ARes = See ACS Signed Content.	For ARes message, see ACS Signed Content.
SDK Maximum Timeout Field Name: sdkMaxTimeout	Indicates maximum amount of time (in minutes) for all exchanges.	3DS SDK	<p>Length: 2 characters</p> <p>JSON Data Type: String</p> <p>Values accepted:</p> <ul style="list-style-type: none"> • Greater than or = 05 	01-APP	01-PA 02-NPA	AReq = R	
SDK Reference Number Field Name: sdkReferenceNumber	Identifies the vendor and version of the 3DS SDK that is utilised for a specific transaction. The value is assigned by EMVCo when the LOA of the specific 3DS SDK is issued.	3DS SDK (sent via 3DS Server)	<p>Length: Variable maximum 32 characters</p> <p>JSON Data Type: String</p>	01-APP	01-PA 02-NPA	AReq = R	

Data Element/ Field Name	Description	Source	Length/Format/Values	Device Channel	Message Category	Message Inclusion	Conditional Inclusion
SDK Server Signed Content Field Name: sdkServerSignedContent	Contains the JWS object (represented as a string) created by the Split-SDK Server for the AReq message. See Section 6.2.2.3 for details.	3DS SDK	Length: Variable JSON Data Type: String Values accepted: <ul style="list-style-type: none">• The body of the JWS object (represented as a string) will contain the following data elements as defined in Table A.1:<ul style="list-style-type: none">○ SDK Reference Number○ SDK Signature Timestamp○ SDK Transaction ID○ Split-SDK Server ID	01-APP	01-PA 02-NPA	AReq = C	Required if SDK Type = 02.
SDK Signature Timestamp Field Name: sdkSignatureTimestamp	Date and time indicating when the 3DS SDK generated the Split-SDK Server Signed Content converted into UTC.	3DS SDK	Length: 14 characters JSON Data Type: String Date format accepted: YYYYMMDDHHMMSS	01-APP	01-PA 02-NPA	See SDK Server Signed Content	See SDK Server Signed Content.

Data Element/ Field Name	Description	Source	Length/Format/Values	Device Channel	Message Category	Message Inclusion	Conditional Inclusion
SDK Transaction ID Field Name: sdkTransID	Universally unique transaction identifier assigned by the 3DS SDK to identify a single transaction.	3DS SDK (sent via 3DS Server)	Length: 36 characters JSON Data Type: String Canonical format as defined in IETF RFC 4122. This may utilise any of the specified versions if the output meets specified requirements.	01-APP	01-PA 02-NPA	AReq = R ARes = R CReq = R CRes = R RReq = R RRes = R Erro = C	Required in Error Message if available (e.g., can be obtained from a message or is being generated).
SDK Type Field Name: sdkType	Indicates the type of 3DS SDK. This data element provides additional information to the DS and ACS to determine the best approach for handling the transaction.	3DS Server	Length: 2 characters JSON Data Type: String Values accepted: <ul style="list-style-type: none">• 01 = Default-SDK• 02 = Split-SDK• 03–79 = Reserved for EMVCo future use (values invalid until defined by EMVCo)• 80–99 = Reserved for DS use	01-APP	01-PA 02-NPA	AReq = R	

Data Element/ Field Name	Description	Source	Length/Format/Values	Device Channel	Message Category	Message Inclusion	Conditional Inclusion
Seller Information Field Name: sellerInfo	Additional transaction information for transactions where merchants submit transaction details on behalf of another entity, i.e., individual sellers in a marketplace or drivers in a ride share platform.	3DS Server	Length: Variable, 1–50 elements JSON Data Type: Array of Objects Refer to Table A.19 for data elements to include.	01-APP 02-BRW 03-3RI	01-PA 02-NPA	AReq = O	

Data Element/ Field Name	Description	Source	Length/Format/Values	Device Channel	Message Category	Message Inclusion	Conditional Inclusion
Serial Number Field Name: serialNum	If present in PReq message, the DS returns Card Range Data that has been updated since the time of the PRes message. If absent, the DS returns all card ranges. If present in the PRes message, indicates the current state of the Card Range Data (the specific value is only meaningful to the DS). The 3DS Server should retain this value for submission in a future PReq message to request only changes that have been made to the Card Range Data since the PRes message was generated. Note: Serial Number is not provided when the DS and the 3DS Server select the Card Range Data File download option.	DS 3DS Server	Length: Variable, maximum 20 alphanumeric characters JSON Data Type: String	N/A	N/A	PReq = O PRes = C	PRes: Absent if the Card Range Data File URL is present.

Data Element/ Field Name	Description	Source	Length/Format/Values	Device Channel	Message Category	Message Inclusion	Conditional Inclusion
SPC Incompletion Indicator Field Name: spcIncompInd	Reason that the SPC authentication was not completed.	3DS Server	Length: 2 characters JSON Data Type: String Values accepted: <ul style="list-style-type: none"> • 01 = SPC did not run or did not successfully complete • 02 = Cardholder cancelled the SPC authentication • 03 = SPC timed out • 04–99 = Reserved for EMVCo future use (values invalid until defined by EMVCo) 	02-BRW	01-PA 02-NPA	AReq = C	Required if the 3DS Requestor attempts to invoke SPC API and there is an error.
SPC Transaction Data Field Name: spcTransData	Information that the 3DS Requestor passes in the SPC API for display in the Smart Modal Window	ACS DS	JSON Data Type: Object Refer to Table A.28 for data elements.	02-BRW	01-PA 02-NPA	ARes= C	Required when the Transaction Status = S

Data Element/ Field Name	Description	Source	Length/Format/Values	Device Channel	Message Category	Message Inclusion	Conditional Inclusion
Split-SDK Server ID Field Name: splitSdkServerID	DS assigned Split-SDK Server identifier. Each DS can provide a unique ID to each Split-SDK Server on an individual basis.	Split-SDK Server	Length: Variable, maximum 32 characters JSON Data Type: String Value accepted: <ul style="list-style-type: none">• Any individual DS may impose specific formatting and character requirements on the contents of this field.	01-APP	01-PA 02-NPA	See SDK Server Signed Content	See SDK Server Signed Content.

Data Element/ Field Name	Description	Source	Length/Format/Values	Device Channel	Message Category	Message Inclusion	Conditional Inclusion
Split-SDK Type Field Name: splitSdkType	<p>Indicates the characteristics of a Split-SDK.</p> <p>Split-SDK Variant: Implementation characteristics of the Split-SDK client</p> <p>Limited Split-SDK Indicator: If the Split-SDK client has limited capabilities</p> <p>Example:</p> <pre>"splitSdkType": { "sdkVariant": "01", "limitedInd": "Y" }</pre>	3DS Server	<p>Length: Variable</p> <p>JSON Data Type: Object sdkVariant</p> <p>Length: 2 characters</p> <p>JSON Data Type: String</p> <p>Values accepted:</p> <ul style="list-style-type: none"> • 01 = Native Client • 02 = Browser • 03 = Shell • 04–79 = Reserved for EMVCo future use (values invalid until defined by EMVCo) • 80–99 = Reserved for DS use <p>limitedInd</p> <p>Length: 1 character</p> <p>JSON Data Type: String</p> <p>Value accepted:</p> <ul style="list-style-type: none"> • Y = Limited <p>Only present if value = Y</p>	01-APP	01-PA 02-NPA	AReq = C	Required if SDK Type = 02

Data Element/ Field Name	Description	Source	Length/Format/Values	Device Channel	Message Category	Message Inclusion	Conditional Inclusion
Submit Authentication Label Field Name: submitAuthenticationLabel	<p>Label to be used in the UI for the button that the user selects when they have completed the authentication.</p> <p>Note: This data element is not used for OOB authentication.</p>	ACS	<p>Length: Variable maximum 45 characters</p> <p>JSON Data Type: String</p>	01-APP	01-PA 02-NPA	CRes = C	Required when the ACS UI Type = 01, 02, or 03.
Tax ID Field Name: taxId	Cardholder's tax identification.	3DS Server	<p>Length: Variable maximum 45 characters</p> <p>JSON Data Type: String</p>	01-APP 02-BRW 03-3RI	01-PA 02-NPA	AReq = C	Conditional based on DS rules.

Data Element/ Field Name	Description	Source	Length/Format/Values	Device Channel	Message Category	Message Inclusion	Conditional Inclusion
Toggle Position Indicator Field Name: togglePositionInd	Indicates if the Trust List and/or Device Binding prompt should be presented below or above the action buttons (Submit Authentication, OOB App, OOB Continuation, Information Continuation, Challenge Additional).	ACS	<p>Length: 2 characters</p> <p>JSON Data Type: String</p> <p>Values accepted:</p> <ul style="list-style-type: none"> • 01 = Above the buttons <p>Only present if value = 01</p> <p>If the Toggle Position Indicator is not present, the Trust List or Device Binding are below the action buttons.</p> <ul style="list-style-type: none"> • 02–99 = Reserved for EMVCo future use (values invalid until defined by EMVCo) 	01-APP	01-PA 02-NPA	CRes = O	

Data Element/ Field Name	Description	Source	Length/Format/Values	Device Channel	Message Category	Message Inclusion	Conditional Inclusion
Transaction Challenge Exemption Field Name: transChallengeExemption	<p>Exemption applied by the ACS to authenticate the transaction without requesting a challenge.</p> <p>Note: The accepted values match the values of the 3DS Requestor Challenge Indicator.</p>	ACS	<p>Length: 2 characters JSON Data Type: String Values accepted:</p> <ul style="list-style-type: none"> • 05 = Transaction Risk Analysis exemption • 08 = Trust List exemption • 10 = Low Value exemption • 11 = Secure Corporate Payments exemption • 79 = No exemption applied • 01–04, 06, 07, 09 and 12–78 = Reserved for EMVCo future use (values invalid until defined by EMVCo) • 80–99 = Reserved for DS use 	01-APP 02-BRW 03-3RI	01-PA 02-NPA	ARes = O	

Transaction Status Field Name: transStatus	Indicates whether a transaction qualifies as an authenticated transaction or account verification. The Final CRes message can only contain a value of Y or N. Transaction Status = C or S is not allowed for Device Channel = 3RI.	ACS DS	Length: 1 character JSON Data Type: String <ul style="list-style-type: none"> • Y = Authentication Verification Successful. • N = Not Authenticated /Account Not Verified; Transaction denied. • U = Authentication/ Account Verification Could Not Be Performed; Technical or other problem, as indicated in ARes or RReq. • A = Attempts Processing Performed; Not Authenticated/ Verified, but a proof of attempted authentication/verification is provided. • C = Challenge Required; Additional authentication is required using the CReq/CRes. • D = Challenge Required; Decoupled Authentication confirmed. • R = Authentication/ Account Verification 	01-APP 02-BRW 03-3RI	01-PA 02-NPA	01-PA: ARes = R RReq = R Final CRes = R 02-NPA: ARes = C RReq = C Final CRes = C	For 01-PA, see Table A.17 for Transaction Status presence conditions. For 02-NPA, requirements for the presence and values of Transaction Status are DS-specific.
---	--	---------------	--	------------------------------------	---------------------	---	--

Data Element/ Field Name	Description	Source	Length/Format/Values	Device Channel	Message Category	Message Inclusion	Conditional Inclusion
			<p>Rejected; Issuer is rejecting authentication/verification and request that authorisation not be attempted.</p> <ul style="list-style-type: none">• I = Informational Only; 3DS Requestor challenge preference acknowledged.• S = Challenge using SPC				

Transaction Status Reason Field Name: transStatusReason	Provides information on why the Transaction Status field has the specified value.	ACS DS	Length: 2 characters JSON Data Type: String Values accepted: <ul style="list-style-type: none"> • 01 = Card authentication failed • 02 = Unknown device • 03 = Unsupported device • 04 = Exceeds authentication frequency limit • 05 = Expired card • 06 = Invalid card number • 07 = Invalid transaction • 08 = No card record • 09 = Security failure • 10 = Stolen card • 11 = Suspected fraud • 12 = Transaction not permitted to Cardholder • 13 = Cardholder not enrolled in service • 14 = Transaction timed out at the ACS • 15 = Low confidence • 16 = Medium confidence 	01-APP 02-BRW 03-3RI	01-PA 02-NPA	ARes = C RReq = C	For 01-PA, required if Transaction Status = N, U, or R. For 02-NPA, requirements for the presence and values of Transaction Status are DS-specific.
---	---	---------------	---	------------------------------------	---------------------	--------------------------	--

			<ul style="list-style-type: none">• 17 = High confidence• 18 = Very high confidence• 19 = Exceeds ACS maximum challenges• 20 = Non-Payment transaction not supported• 21 = 3RI transaction not supported• 22 = ACS technical issue• 23 = Decoupled Authentication required by ACS but not requested by 3DS Requestor• 24 = 3DS Requestor Decoupled Max Expiry Time exceeded• 25 = Decoupled Authentication was provided insufficient time to authenticate Cardholder. ACS will not make attempt• 26 = Authentication attempted but not performed by the Cardholder• 27 = Preferred Authentication Method not supported			
--	--	--	--	--	--	--

Data Element/ Field Name	Description	Source	Length/Format/Values	Device Channel	Message Category	Message Inclusion	Conditional Inclusion
			<ul style="list-style-type: none"> • 28 = Validation of content security policy failed • 29 = Authentication attempted but not completed by the Cardholder. Fall back to Decoupled Authentication • 30 = Authentication completed successfully but additional authentication of the Cardholder required. Reinitiate as Decoupled Authentication • 31–79 = Reserved for EMVCo future use (values invalid until defined by EMVCo) • 80–99 = Reserved for DS use 				
Transaction Status Reason Information Field Name: transStatusReasonInfo	Provides additional information on the Transaction Status Reason.	ACS DS	Length: Variable, maximum 256 characters JSON Data Type: String	01-APP 02-BRW 03-3RI	01-PA 02- NPA	ARes = O RReq = O	

Data Element/ Field Name	Description	Source	Length/Format/Values	Device Channel	Message Category	Message Inclusion	Conditional Inclusion
Transaction Type Field Name: transType	Identifies the type of transaction being authenticated.	3DS Server	Length: 2 characters JSON Data Type: String Values accepted: <ul style="list-style-type: none"> • 01 = Goods/ Service Purchase • 03 = Check Acceptance • 10 = Account Funding • 11 = Quasi-Cash Transaction • 28 = Prepaid Activation and Load Note: Values derived from ISO 8583-1.	01-APP 02-BRW 03-3RI	01-PA	AReq = C	This field is required in some markets (e.g., for Merchants in Brazil). Otherwise, optional.
Trust List Data Entry Field Name: trustListDataEntry	Indicator provided by the 3DS SDK to the ACS to confirm whether the Cardholder gives consent to Trust List.	3DS SDK	Length: 1 character JSON Data Type: String Values accepted: <ul style="list-style-type: none"> • Y = Consent given to Trust List • N = Consent not given to Trust List Note: If the Cardholder action changes the default value, then the value = Y. Otherwise the value = N.	01-APP	01-PA 02-NPA	CReq = C	Required if Trust List Information Text was present in the preceding CRes message

Data Element/ Field Name	Description	Source	Length/Format/Values	Device Channel	Message Category	Message Inclusion	Conditional Inclusion
Trust List Information Text Field Name: trustListInfoText	<p>Text provided by the ACS to the Cardholder during a Trust List transaction.</p> <p>Example:</p> <ul style="list-style-type: none"> “Would you like to add this Merchant to your Trust List?” 	ACS	Length: Variable, maximum 64 characters	01-APP	01-PA 02-NPA	CRes = O	See Table A.20 for presence conditions.
Trust List Status Field Name: trustListStatus	Enables the communication of trust list status between the ACS, the DS and the 3DS Requestor.	3DS Server DS ACS	<p>Length: 1 character JSON Data Type: String Values accepted:</p> <ul style="list-style-type: none"> Y = 3DS Requestor is Trust Listed by Cardholder N = 3DS Requestor is not Trust Listed by Cardholder E = Not eligible as determined by issuer P = Pending confirmation by Cardholder R = Cardholder rejected U = Trust List status unknown, unavailable, or does not apply <p>Note: Valid values in the AReq message are Y or N</p>	01-APP 02-BRW 03-3RI	01-PA 02-NPA	AReq = O ARes = O RReq = O	

Data Element/ Field Name	Description	Source	Length/Format/Values	Device Channel	Message Category	Message Inclusion	Conditional Inclusion
Trust List Status Source Field Name: trustListStatusSource	This data element will be populated by the system setting Trust List Status.	3DS Server DS ACS	Length: 2 characters JSON Data Type: String Values accepted: <ul style="list-style-type: none">• 01 = 3DS Server• 02 = DS• 03 = ACS• 04-79 = Reserved for EMVCo future use (values invalid until defined by EMVCo)• 80-99 = Reserved for DS use	01-APP 02-BRW 03-3RI	01-PA 02-NPA	AReq = C ARes = C RReq = C	Required if Trust List Status is present.

Data Element/ Field Name	Description	Source	Length/Format/Values	Device Channel	Message Category	Message Inclusion	Conditional Inclusion
WebAuthn Credential List Field Name: webAuthnCredList	List of credential IDs registered for the Cardholder Account Number.	ACS	<p>Size: Variable, 1–10 elements JSON Data Type: Array of Objects The object contains:</p> <ul style="list-style-type: none"> • Relying Party ID Field Name: rPID Length: Variable, maximum 2048 characters • WebAuthn Credential Field Name: credentialIds Length: Variable, 16–1000 characters Base64url-encoded 	02-BRW	01-PA 02-NPA	ARes = C	Required when Transaction Status = S
Why Information Label Field Name: whyInfoLabel	Label to be displayed to the Cardholder for the "why" information section.	ACS	<p>Length: Variable, maximum 45 characters JSON Data Type: String</p>	01-APP	01-PA 02-NPA	CRes = O	

Data Element/ Field Name	Description	Source	Length/Format/Values	Device Channel	Message Category	Message Inclusion	Conditional Inclusion
Why Information Text Field Name: whyInfoText	Text provided by the Issuer to be displayed to the Cardholder to explain why the Cardholder is being asked to perform the authentication task.	ACS	Length: Variable, maximum 256 characters JSON Data Type: String Note: Carriage return is supported in this data element and is represented by an “\n”. Note: Bold text is supported in this data element and is enclosed between **. For example “This is **bold** text” is rendered as This is bold text.	01-APP	01-PA 02-NPA	CRes = O	

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

A.5 Device Information—01-APP Only

The Device Information is gathered by the 3DS SDK as defined in the *EMV 3-D Secure SDK—Device Information* specification. This data is placed into a JWE object that will encrypt the data using the DS public key.

A.6 Browser Information—02-BRW Only

Accurate Browser Information is obtained in the AReq message for an ACS to determine the ability to support authentication on a particular Cardholder Browser for each transaction. The 3DS Server needs to accurately populate the Browser information for each transaction. This data

can be obtained by 3DS software provided to the 3DS Requestor or through for example, remote JavaScript calls. It is the responsibility of the 3DS Server to ensure that the data is not altered or hard-coded and that it is unique to each transaction. The specific fields captured from the Cardholder Browser for each transaction are:

- Browser Accept Headers
- Browser IP Address
- Browser Java Enabled
- Browser Language
- Browser Screen Color Depth
- Browser Screen Height
- Browser Screen Width
- Browser Time Zone
- Browser User-Agent

Refer to Table A.1 for data element specifications.

A.7 3DS Method Data

The following table defines the data elements sent in the 3DS Method. The data is exchanged between the 3DS Requestor via the Cardholder Browser. The HTTP field name is `threeDSMethodData`.

Table A.2: 3DS Method Data

Data Element/Field Name	Description	Length/Format/Values	Recipient	Message Category	Message Inclusion
3DS Method Notification URL Field Name: threeDSMethodNotificationURL	The URL that will receive the notification of 3DS Method completion from the ACS. This is sent in the initial request to the ACS from the 3DS Requestor executing the 3DS Method.	Length: Variable, Maximum 2048 characters JSON Data Type: String Value accepted: <ul style="list-style-type: none">• Fully Qualified URL	ACS	01-PA 02-NPA	R
3DS Server Transaction ID Field Name: threeDSServerTransID	A unique identifier for the transaction that will be the same as the 3DS Server Transaction ID in the AReq message and will have the same format as specified in Table A.1. This will be sent to the ACS in the 3DS Method HTTP POST and will be returned in the POST to the 3DS Method Notification URL.	Refer to 3DS Server Transaction ID in Table A.1.	ACS 3DS Requestor	01-PA 02-NPA	R

3DS Method Data Examples

- **Example 1:** threeDSMethodData to be sent to ACS in the 3DS Method HTTP form POST from 3DS Requestor

```
<form name="frm" method="POST" action="Rendering URL">  
<input type="hidden" name="threeDSMethodData"  
value="eyJ0aHJlZURTU2VydmVyVHJhbnNJRCI6IjNhYzdjYWE3LWFhNDItMjY2My03OTFiLTJhYzA1YTU0MmM0YSIsInRocmVlRFNNZX  
Rob2ROb3RpZmljYXRpb25VUkwiOiJ0aHJlZURTTWV0aG9kTm90aWZpY2F0aW9uVVJMIn0">  
</form>
```

Decoded threeDSMethodData:

```
{"threeDSServerTransID": "3ac7caa7-aa42-2663-791b-  
2ac05a542c4a", "threeDSMethodNotificationURL": "threeDSMethodNotificationURL"}
```

- **Example 2:** threeDSMethodData to be sent to 3DS Method Notification URL from the ACS

```
<form name="frm" method="POST" action="threeDSMethodNotificationURL">  
<input type="hidden" name="threeDSMethodData"  
value="eyJ0aHJlZURTU2VydmVyVHJhbnNJRCI6IjNhYzdjYWE3LWFhNDItMjY2My03OTFiLTJhYzA1YTU0MmM0YSJ9">  
</form>
```

Decoded threeDSMethodData:

```
{"threeDSServerTransID": "3ac7caa7-aa42-2663-791b-2ac05a542c4a"}
```

A.8 Browser CReq and CRes POST

The following table defines the data elements sent in the Browser POST to the ACS for the CReq flow, and to the Notification URL in the CRes flow. An HTML form is utilised within the Cardholder Browser and the data is redirected via the Cardholder Browser in an HTTP POST.

Note: The end result of the redirection must be similar as if an HTML tag was utilised.

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

Data Element / Field Name	Description	Recipient	Length/Format/Values	Message Inclusion
3DS Requestor Session Data Field Name: threeDSsessionData	<p>The 3DS Requestor may provide the 3DS Requestor Session Data with the CReq message to the ACS. The 3DS Requestor Session Data is optionally used to accommodate the different methods that 3DS Requestor systems use to handle session information.</p> <p>The ACS returns the 3DS Requestor session data with the CRes message POST to the 3DS Requestor. If the 3DS Requestor system can associate the final post with the original session without further assistance, the 3DS Requestor Session Data field may be missing.</p> <p>If the 3DS Requestor system does not maintain a session for a given authentication session, the 3DS Requestor Session Data field can carry any data the 3DS Requestor needs to continue the session.</p> <p>Because the content of this field varies by 3DS Requestor implementation, the ACS preserves the content unchanged and without assumptions.</p> <p>If provided by the 3DS Requestor, the Session Data must be returned by the ACS.</p>	ACS 3DS Requestor	Length: Maximum 1024 Format: Alphanumeric Base64url-encoded The size of the field (after Base64url-encoding, if applicable) is limited to 1024 bytes.	<ul style="list-style-type: none"> O in HTML form with the CReq message R in HTML form with the CRes message if received with the CReq message
CReq Field Name: creq	The entire CReq message, as defined in Table B.3, that has been Base64url-encoded.	ACS	Length: Variable, Base64url-encoded	R
CRes Field name: cres	The entire CRes message, as defined in Table B.4, that has been Base64url-encoded.	3DS Requestor		R

Browser CReq - CRes Data Examples

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

```
3DS Requestor Session Data from the 3DS Requestor = "merchant.com-ID-adac2434-df78-4bfa-bcd9-11ca4cccd5dca"
```

```
3DS Requestor Session Data base64URL encoded =
"bwVY2hhbnQuY29tLU1ELWFkYWMMyNDM0LWRmNzgtNGJmYS1iY2Q5LTExy2E0Y2NkNWRjYQ"
```

CReq message

```
{
    "threeDSServerTransID": "8a880dc0-d2d2-4067-bcb1-b08d1690b26e",
    "acsTransID": "d7c1ee99-9478-44a6-b1f2-391e29c6b340",
    "threeDSRequestorURL": "https://merchant.com/url",
    "messageType": "CReq",
    "messageVersion": "2.3.0"
}
```

CReq message base64URL encoded

```
"eyJ0aHJlZURTU2VydmVyVHJhbNJRCl6IjhODgwZGMwLWQyZDItNDA2Ny1iY2IxLWIwOGQxNjkwYjI2ZSIscSJhY3NUcmFuc01EIjoiZDdjMWVlOTktOTQ3OC00NGE2LWIxZjItMzkxZTI5YzZiMzQwIiwidGhyZWVEU1JlcXVlc3Rvc1VybCI6Imh0dHBzOi8vbWVY2hhbnQuY29tL3VybCIsIm1lc3NhZ2VUeXB1IjoiQ1JlcSIsIm1lc3NhZ2VWZXJzaW9uIjoiMi4zLjAifQ"
```

HTML form

```
"htmlCreq": "<form action='https://acs.com.creq' method='post'>
<input type='hidden' name='creq' value='
'eyJ0aHJlZURTU2VydmVyVHJhbNJRCl6IjhODgwZGMwLWQyZDItNDA2Ny1iY2IxLWIwOGQxNjkwYjI2ZSIscSJhY3NUcmFuc01EIjoiZDdjMWVlOTktOTQ3OC00NGE2LWIxZjItMzkxZTI5YzZiMzQwIiwidGhyZWVEU1JlcXVlc3Rvc1VybCI6Imh0dHBzOi8vbWVY2hhbnQuY29tL3VybCIsIm1lc3NhZ2VUeXB1IjoiQ1JlcSIsIm1lc3NhZ2VWZXJzaW9uIjoiMi4zLjAifQ' />
```

```
<input type=\\"hidden\\" name=\\"threeDSSessionData\\" value=\\"b\\' bWVY2hhbnQuY29tLU1ELWFkYWMyNDM0LWRmNzgtNGJmYS1iY2Q5LTExY2E0Y2NkNWRjYQ\\' \\" /></form>"
```

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

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

CRes message

```
{  
    "threeDSServerTransID": "8a880dc0-d2d2-4067-bcb1-b08d1690b26e",  
    "acsTransID": "d7c1ee99-9478-44a6-b1f2-391e29c6b340",  
    "transStatus": "Y",  
    "messageType": "CRes",  
    "messageVersion": "2.3.0"  
}
```

Base64 URL encoded CRes message

```
"eyJ0aHJlZURTU2VydmVyVHJhbNJRCl6IjhODgwZGMwLWQyZDItNDA2Ny1iY2IxLWIwOGQxNjkWYjI2ZSIImFjc1RyYW5zSUQiOijK  
N2MxZWU5OS05NDc4LTQ0YTytYjFmMi0zOTFlMjljNmIzNDAlLCJ0cmFuc1N0YXR1cyI6IlkiLCJtZXNzYWdlVHlwZSI6IkNSZXMiLCJtZ  
XNzYWdlVmVyc2lvbiI6IjIuMy4wiix9"
```

3DS Requestor Session Data base64URL encoded =
"bWVY2hhbnQuY29tLU1ELWFkYWMyNDM0LWRmNzgtNGJmYS1iY2Q5LTExY2E0Y2NkNWRjYQ"

HTML form

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

```
<input type='hidden' name='cres' value='eyJ0aHJlZURTU2VydmVyVHJhbnNJRCl6IjhODgwZGMwLWQyZDItNDA2Ny1iY2IxLWIwOGQxNjkwyjI2ZSIsImFjc1RyYW5zsUQiOijK2MxZWU5OS05NDc4LTQ0YTytYjFmMi0zOTF1MjljNmIzNDAiLCJ0cmFuc1N0YXR1cyI6IlkilCJtZXNzYWdlVHlwZSI6IkNSZXMiLCJtZXNzYWdlVmVyc2lvbiI6IjIuMy4wIix9' />
<input type='hidden' name='threeDSSessionData' value='b'b' bWVyY2hhbnQuY29tLU1ELWFkYWMYNDM0LWRmNzgtNGJmYS1iY2Q5LTExy2E0Y2NkNWRjYQ' /></form>"
```

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

Error messages are used to determine how to formulate a response when a system receives a message that cannot be processed, or when the error is required as part of receiving a response message from another system.

For example, a 3DS Server receives an ARes message from a DS that contains an error, and the 3DS Server responds with an Error message to the DS using the 3DS Server Transaction ID of the transaction that had an error.

The following table identifies the Error Code values and specifies the associated content for Error Description and Error Detail. The information provided in the Error Description and Error Detail are guidelines on the expected content. In Error Detail if there is a mention of listing required elements, the expectation is that those data elements will be listed appropriately.

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

Value	Error Code	Error Description	Error Detail
101	Message Received Invalid	<p>One of the following:</p> <ul style="list-style-type: none"> • Message is not AReq, ARes, CReq, CRes, PReq, PRes, OReq, ORes, RReq, or RRes. • Valid Message Type is sent to or from an inappropriate component (such as AReq message being sent to the 3DS Server). • Message not recognised. 	<p>One of the following:</p> <ul style="list-style-type: none"> • Invalid Message Type • Invalid Message for the receiving component • Invalid Formatted Message
102	Message Version Number Not Supported	<p>One of the following:</p> <ul style="list-style-type: none"> • Message Version Number received is not valid for the receiving component. • Error in the Message Version Number in the Card Range Data. • Message Version Number provided for a Payment Token is not supported by the actual PAN when de-tokenised. 	<ul style="list-style-type: none"> • Message Version Number in the Card Range Data is not active. • Message Version Number received is not supported by the receiving component. <p>Note: All supported Protocol Version Numbers are provided in a comma delimited list.</p>
103	Sent Messages Limit Exceeded	Exceeded maximum number of PReq messages sent to the DS.	For example, the 3DS Server sends two PReq messages to the DS within one hour.
201	Required Data Element Missing	A message element required as defined in Table A.1 is missing from the message.	<p>Name of required element(s) that was omitted; if more than one element is detected, this is a comma delimited list.</p> <p>Parent Example: <code>messageType</code></p> <p>Parent/Child Example: <code>acctInfo.chAccAgeInd</code></p>

Value	Error Code	Error Description	Error Detail
202	Critical Message Extension Not Recognised	Critical message extension not recognised.	ID of critical Message Extension(s) that was not recognised; if more than one extension is detected, this is a comma delimited list of message identifiers that were not recognised.
203	Format or value of one or more Data Elements is Invalid according to the Specification	<ul style="list-style-type: none"> • Data element not in the required format or value is invalid as defined in Table A.1, or • Message Version Number does not match the value set by the 3DS Server in the AReq message, or • Data element is present in a message where the conditional inclusion does not apply. • UTC date and time data element is not using UTC. 	Name of invalid element(s); if more than one invalid data element is detected, this is a comma delimited list.
204	Duplicate Data Element	Valid data element presents more than once in the message.	Name of duplicated data element; if more than one duplicate data element is detected, this is a comma delimited list.
205	Card Range Overlap	<p>Overlap in the card ranges provided by the DS in the PRes message.</p> <p>For example, the two card ranges 11000–15000 and 13000–17000 overlap from 13000–15000.</p>	List of card Ranges that overlap.
206	Card Range Action Indicator	<p>Action is not possible for the card range.</p> <p>For example, Delete or Modify a card range that does not exist, or Add an already existing card range.</p>	List the Card Range and Action Indicator that is causing the error.
207	Value in the Reserved Value range	Data Element value is in the range of “Reserved for DS use” or “Reserved for EMVCo future use” and is not recognised.	Name of invalid element(s); if more than one invalid data element is detected, this is a comma delimited list.

Value	Error Code	Error Description	Error Detail
301	Transaction ID Not Recognised	Transaction ID received is not valid for the receiving component.	The Transaction ID received was invalid. Invalid meaning Transaction ID not recognised.
302	Data Decryption Failure	Data could not be decrypted by the receiving system due to technical or other reason.	Description of the failure.
303	Access Denied, Invalid Endpoint	Access denied, invalid endpoint.	Description of the failure.
304	ISO Code Invalid	ISO code not valid per ISO tables (for either country or currency), or code is one of the excluded values listed in Table A.5.	Name of invalid element(s); if more than one invalid element is detected this is a comma delimited list. If Challenge Request.Purchase.currency and Challenge Request.Purchase.exponent form an invalid pair, list both as Error Description.
305	Transaction Data Not Valid	If in response to an AReq message: <ul style="list-style-type: none">• Cardholder Account Number is not in a range belonging to Issuer If in response to a CReq, and a CReq message was incorrectly sent: <ul style="list-style-type: none">• CReq message was received by the wrong ACS, OR• CReq message was not sent, based on the values in the ARes message	Name of element(s) that caused the ACS to decide that the AReq message or CReq message was incorrectly sent; if more than one invalid element is detected this is a comma-delimited list.
306	Merchant Category Code (MCC) Not Valid for Payment System	Merchant Category Code (MCC) not valid for Payment System.	For example, invalid MCC received in the AReq message.
307	Serial Number Not Valid	Serial Number not valid.	For example, invalid Serial Number in the PReq/PRes message (e.g., too old, not found).

Value	Error Code	Error Description	Error Detail
308	Signature Verification Failure	SDK Server Signed Content could not be verified.	Description of the failure.
309	Validation against content security policies Failure	Validation against content security policies failed.	For example, which element prevented successful validation.
310	Incorrect Cryptographic Algorithm	The use of a specific cryptographic algorithm is not allowed in the specific context.	For example, which cryptographic algorithm was expected.
311	Incorrect kid	The DS detects an error for the key identifier (kid) present in the SDK Encrypted Data.	For example: <ul style="list-style-type: none"> • The provided kid is not recognised • The kid is not present
312	Duplicate message	A message with the same Transaction ID was already received.	The Transaction ID is recognised as a duplicate. For example, the DS receives multiple RReq messages with the same Transaction ID.
313	Inconsistent RReq message	An RReq message is received although there was no challenge (Transaction Status not equal to C or D) for this transaction.	The ACS sends an RReq message but the Transaction Status in the corresponding ARes message was not = C or D.
314	Multiple CReq messages not supported	During a challenge for the Browser flow, the ACS does not accept multiple CReq messages.	The Cardholder requests a Browser page refresh during a challenge, the 3DS Server sends a second CReq message to the ACS.
315	CReq message received after the RReq message	During a challenge for the Browser flow, the ACS receives a CReq message, after having sent the RReq message.	The Cardholder requests a Browser page refresh during a challenge after the ACS has sent the RReq message to complete the transaction.
402	Transaction Timed Out	Transaction timed-out.	For example, Timeout expiry reached for the transaction as defined in Section 5.5.

Value	Error Code	Error Description	Error Detail
403	Transient System Failure	Transient system failure.	For example, a slowly processing back-end system.
404	Permanent System Failure	Permanent system failure.	For example, a critical database cannot be accessed.
405	System Connection Failure	System connection failure.	For example, the sending component is unable to establish connection to the receiving component.

A.10 Excluded ISO Currency and Country Code Values

The following table lists exclusions from the ISO values for Currency Code (ISO 4217) and Country Code (ISO 3166).

Table A.5: Excluded Currency Code and Country Code Values

ISO Code	Value Not Permitted for 3-D Secure	Definition
ISO 4217	955	European Composite Unit
ISO 4217	956	European Monetary Unit
ISO 4217	957	European Unit of Account 9
ISO 4217	958	European Unit of Account 17
ISO 4217	959	Gold
ISO 4217	960	I.M.F.
ISO 4217	961	Silver
ISO 4217	962	Platinum
ISO 4217	963	Reserved for testing
ISO 4217	964	Palladium
ISO 4217	999	No currency is involved
ISO 3166-1	901–999	Reserved by ISO to designate country names not otherwise defined

A.11 Card Range Data

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

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

Table A.6: Card Range Data

Data Element/Field Name	Description	Length/Format/Values	Inclusion
Ranges Field Name: ranges	<p>The Ranges array contains the Start Range and End Range. It contains one or more card ranges.</p> <p>Refer to the following elements:</p> <ul style="list-style-type: none">• Start Range• End Range	<p>Size: Variable, maximum 1–5,000 elements</p> <p>JSON Data Type: Array of Object</p>	R
Start Range Field Name: start	Start of the card range.	<p>Length: 13–19 characters</p> <p>JSON Data Type: String</p>	R

Data Element/Field Name	Description	Length/Format/Values	Inclusion
End Range Field Name: end	End of the card range.	Length: 13–19 characters JSON Data Type: String	R
Action Indicator Field Name: actionInd	Indicates the action to take with the card range. Note: M (Modify the card range data) is used only to modify or update data associated with the card ranges, not to modify the Start Range and End Range.	Length: 1 character JSON Data Type: String Values accepted: <ul style="list-style-type: none">• A = Add the card range to the cache (default value)• D = Delete the card range from the cache• M = Modify the card range data	O
Issuer Country Code Field Name: issuerCountryCode	Qualify the Issuer country for the Ranges.	Length: 3 characters JSON Data Type: String Values accepted: <ul style="list-style-type: none">• ISO 3166-1 numeric three-digit country code, other than exceptions listed in Table A.5.	O
DS Protocol Versions Field Name: dsProtocolVersions	Contains the list of active protocol versions supported by the DS. If the DS Protocol Version is present in the Card Range data element, it overrides the DS Protocol Versions in the PRes message.	Size: Variable, 1–10 elements JSON Data Type: Array of String String: 5–8 characters Values accepted: <ul style="list-style-type: none">• Refer to <i>EMV Specification Bulletin 255</i>	O

Data Element/Field Name	Description	Length/Format/Values	Inclusion
ACS Protocol Versions Field Name: acsProtocolVersions	<p>Array of objects containing the list of protocol versions supported by the ACS for the card range, with their associated ACS Information Indicator, the 3DS Method URL and the list of Supported Message Extension.</p> <ul style="list-style-type: none"> • Version • ACS Information Indicator • 3DS Method URL • Supported Message Extension 	<p>Size: Variable maximum 1–10 elements JSON Data Type: Array of Objects Values accepted:</p> <ul style="list-style-type: none"> • Refer to the data elements: <ul style="list-style-type: none"> ◦ Version ◦ ACS Information Indicator ◦ 3DS Method URL ◦ Supported Message Extension 	R
Version Field Name: version	The Protocol Version supported by the ACS for the card range.	<p>Length: 5–8 characters JSON Data Type: String Values accepted:</p> <ul style="list-style-type: none"> • Refer to <i>EMV Specification Bulletin 255</i> 	R

Data Element/Field Name	Description	Length/Format/Values	Inclusion
ACS Information Indicator Field Name: acsInfoInd	<p>Provides additional information for a particular protocol version to the 3DS Server. The element lists all applicable values for the card range.</p> <p>Example:</p> <pre>{ "acsInfoInd": ["01", "02", "03", "04", "05", "06", "07"] }</pre>	<p>Size: Variable, maximum 1–99 elements JSON Data Type: Array of String. String: 2 characters. Values accepted:</p> <ul style="list-style-type: none"> • 01 = Authentication Available at ACS • 02 = Attempts Supported by ACS or DS • 03 = Decoupled Authentication Supported • 04 = Trust List Supported • 05 = Device Binding Supported • 06 = WebAuthn Authentication Supported • 07 = SPC Authentication Supported • 08 = Transaction Risk Analysis Exemption Supported • 09 = Trust List Exemption Supported • 10 = Low Value Exemption Supported • 11 = Secure Corporate Payments Exemption Supported • 12–79 = Reserved for EMVCo future use (values invalid until defined by EMVCo) • 80–99 = Reserved for DS use 	O
3DS Method URL Field name: threeDSMethodURL	<p>The ACS URL that will be used by the 3DS Method for a particular protocol version.</p> <p>Note: The <code>threeDSMethodURL</code> data element may be omitted if not supported by the ACS for this specific card range.</p>	<p>Length: Variable, Maximum 2048 characters JSON Data Type: String Value accepted:</p> <ul style="list-style-type: none"> • Fully Qualified URL 	O

Data Element/Field Name	Description	Length/Format/Values	Inclusion
Supported Message Extension Field Name: supportedMsgExt	List of message extensions supported by the ACS that contains the Assigned Extension Group Identifier and the Extension Version Number.	Size: Variable maximum 1–10 elements JSON Data Type: Array of Objects Value accepted: <ul style="list-style-type: none">• Refer to Table A.8<ul style="list-style-type: none">○ Assigned Extension Group Identifier○ Extension Version Number	C Present if not empty

Card Range Data Example

```
{"cardRangeData": [  
    {"ranges": [  
        {"start": "1000000000000000",  
         "end": "100000000005000"},  
        {"start": "100000000006000",  
         "end": "100000000007000"}  
    ],  
    "actionInd": "A",  
    "issuerCountryCode": "356",  
    "dsProtocolVersions": ["2.2.0", "2.3.0", "2.3.1"],  
    "acsProtocolVersions": [  
        {"version": "2.2.0",  
         "acsInfoInd": ["01", "02"],  
         "threeDSMethodURL": "https://www.acs.com/script1"},  
        {"version": "2.3.0",  
         "acsInfoInd": ["01", "02"],  
         "threeDSMethodURL": "https://www.acs.com/script2"},  
        {"version": "2.3.1",  
         "acsInfoInd": ["01", "02"],  
         "threeDSMethodURL": "https://www.acs.com/script3"}  
    ]  
}
```

```
        {"id": "A000000802-001", "version": "2.0"},  
        {"id": "A000000802-004", "version": "1.0"}  
    ],  
    {"version": "2.3.0",  
     "acsInfoInd": ["01", "02", "03", "04", "80"],  
     "threeDSMethodURL": "https://www.acs.com/script2"  
    },  
    {"version": "2.3.1",  
     "acsInfoInd": ["01", "02", "03", "04", "81"],  
     "threeDSMethodURL": "https://www.acs.com/script3"  
    }  
  ]  
}  
]
```

The DS URL List data element contains information returned in a PRes message to the 3DS Server from the specific DS that Contains the list of URLs that the 3DS Server can use to communicate with a DS. Its JSON Data Type: Array of Object contains:

- the 3DS Server to DS URL
- the DS Country Code (optional)

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

Table A.7: DS URLs

Data Element/Field Name	Description	Length/Format/Values	Inclusion
3DS Server to DS URL Field Name: threeDSServerToDsUrl	URL that the 3DS Server uses to communicate with a DS for a particular card range. If the DS Country Code is absent, the 3DS Server can use this URL for all card ranges.	Length: Variable, Maximum 2048 characters JSON Data Type: String Value accepted: <ul style="list-style-type: none">• Fully Qualified URL	O
DS Country Code Field Name: dsCountryCode	Qualify the country for which the 3DS Server to DS URL can be used. For a Card Range Data, if the Issuer Country Code matches the DS Country Code, the 3DS Server uses this 3DS Server to DS URL to communicate with the DS. If there is no match, the 3DS Server uses the default 3DS Server to DS URL.	Length: 3 characters JSON Data Type: String Values accepted: <ul style="list-style-type: none">• ISO 3166-1 numeric three-digit country code, other than exceptions listed in Table A.5.	O

DS URL List Data Example

```
{"dsUrlList": [
    {"dsCountryCode": "356",
     "threeDSServerToDsUrl": "https://www.india-ds.com/3ds/",
     "threeDSServerToDsUrl": "https://www.ds.com/3ds/"}
]
```

A.11.1 Supported Message Extension Data Element

The Supported Message extension data element contains information about the message extension and message extension version that a specific ACS supports. Its JSON Data Type: Array of Object contains:

- the Assigned Extension Group Identifier
- the Extension Version Number

The detailed data elements are outlined in Table A.8

Table A.8: Supported Message Extension

Data Element/Field Name	Description	Length/Format/Values	Inclusion
Assigned Extension Group Identifier Field Name: <code>id</code>	A unique identifier for the extension.	Length: 14 characters JSON Data Type: String Values accepted: <ul style="list-style-type: none">• Refer to <i>EMV Specification Bulletin 255</i>	R
Extension Version Number Field Name: <code>version</code>	Version number of the message extension.	Length: 3 characters JSON Data Type: String Value accepted: <ul style="list-style-type: none">• Refer to the Extension Version Number in the message extension with the corresponding Assigned Extension Group Identifier	R

Supported Message Extension Data Example

```
{"supportedMsgExt": [  
    {"id": "A000000802-001", "version": "2.0"},  
    {"id": "A000000802-004", "version": "1.0"}  
]
```

A.12 Message Extension Data

Message Extensions are used to carry additional data that is not defined in this *Core Specification*. The party defining the Message Extension shall define the format of the data. Examples of data to be sent via extensions:

- Data represented in JSON Objects
- Binary data
- Single data elements

Data shall be sent in the Message Extension field with the data populated within a JSON array. Multiple extensions represented as JSON Objects may be within the JSON array if required. A maximum of 15 extensions (objects) are supported within the Message Extension data element, totalling a maximum of 81920 characters.

The specific elements that shall comprise the extension are:

- Extension name (`name`)
- Assigned extension group identifier (`id`)
- Criticality indicator (`criticalityIndicator`)
- Data (`data`)

For example (with multiple extensions defined):

```
{"messageExtension":  
[  
    {  
        "name": "extension1",  
        "id": "ID1",  
        "criticalityIndicator": true,  
        "data": {  
            "valueOne": "value"  
        }  
    },  
    {  
        "name": "extension2",  
        "id": "ID2",  
        "criticalityIndicator": true,  
        "data": {  
            "valueOne": "value1",  
            "valueTwo": "value2"  
        }  
    },  
    {  
        "name": "sharedData",  
        "id": "ID3",  
        "criticalityIndicator": false,  
    }]
```

```

    "data": {
        "value3": "IkptTT05EYXRhIjogew0KImRhGExIjogInNvbWUgZGF0YSIsDQoizGF0YTII0iAic29tZSBvdGhlciB
kYXRhIg0KfQ=="
    }
}
]
}

```

A.12.1 Message Extension Attributes

Table A.9: Message Extension Attributes

Attribute Name	Description	Length/Format/Value	Inclusion
criticalityIndicator	A Boolean value indicating whether the recipient must understand the contents of the extension to interpret the entire message.	JSON Data Type: Boolean Values accepted: <ul style="list-style-type: none">• true• false	R
data	The data carried in the extension.	Length: Variable, Maximum 8059 characters JSON Data Type: Object	R
id	A unique identifier for the extension. Note: Payment System Registered Application Provider Identifier (RID) is required as prefix of the ID.	Length: Variable, Maximum 64 characters JSON Data Type: String	R
name	The name of the extension data set as defined by the extension owner.	Length: Variable, Maximum 64 characters JSON Data Type: String	R

A.12.2 Identification

Each Message Extension defined for use in 3-D Secure must have a unique identifier assigned. Examples of unique identifiers include:

- EMVCo-assigned IDs
- Object IDs (OID)
- Uniform Resource Identifiers (URI)
- DS-assigned IDs

The party defining the message extension specifies the format of the identifier and the value.

A.12.3 Criticality

The data in a Message Extension may affect the meaning of the rest of the data such that the entire message can only be understood in the context of the extension data. When this occurs, the extension is deemed to be critical and the value of the criticality attribute must = true. When an extension is critical, recipients of the message must recognise and be able to process the extension. If a 3-D Secure application receives a message containing a critical extension that it does not recognise, it must treat the message as invalid and return Error Code = 202. When an extension is non-critical, recipients that cannot recognise the extension must ignore the data and pass it to the destination system unaltered.

All critical Message Extensions shall be assigned by EMVCo.

A.13 3DS Requestor Risk Information

3DS Requestor Risk Information are specific data elements within the AReq message that the 3DS Requestor provides to the ACS in support of the ACS risk assessment. The data elements are optional in the AReq message. However, the presence of the data elements in the AReq message will make the risk-based authentication more precise. By evaluating these data elements, the ACS has data available that can reduce the number of unnecessary challenges.

The data elements include the following types of information:

- **Cardholder Account**—Cardholder's account at the 3DS Requestor (if Cardholder is not a Guest)
- **Merchant Risk Indicator**—Purchase and its risk
- **3DS Requestor Authentication**—How the 3DS Requestor authenticated the Cardholder

- **3DS Requestor Prior Transaction Authentication**—How the 3DS Requestor previously used 3DS to authenticate the Cardholder

These data elements are included in Table A.1 as individual data elements with a JSON Object format. The following sections provide detailed information of each name/value pair (NVP) data element.

A.13.1 Cardholder Account Information

The Cardholder Account Information contains optional information about the Cardholder Account. The detailed data elements, which are optional are outlined in Table A.10.

Note: Cardholder Account Information data elements used to define a time period can be included as either: the specific date or an approximate indicator for when the action occurred. 3DS Requestors can use either format.

Table A.10: Cardholder Account Information

Data Element/Field Name	Description	Length/Format/Values
Cardholder Account Age Indicator Field Name: chAccAgeInd	Length of time that the cardholder has had the account with the 3DS Requestor.	Length: 2 characters JSON Data Type: String Values accepted: <ul style="list-style-type: none">• 01 = No account (guest checkout)• 02 = Created during this transaction• 03 = Less than 30 days• 04 = 30–60 days• 05 = More than 60 days
Cardholder Account Change Field Name: chAccChange	Date converted into UTC that the Cardholder's account with the 3DS Requestor was last changed, including Billing or Shipping address, new payment account, or new user(s) added.	Length: 8 characters JSON Data Type: String Format accepted: <ul style="list-style-type: none">• Date format = YYYYMMDD

Data Element/Field Name	Description	Length/Format/Values
Cardholder Account Change Indicator Field Name: chAccChangeInd	Length of time since the Cardholder's account information with the 3DS Requestor was last changed, including Billing or Shipping address, new payment account, or new user(s) added.	Length: 2 characters JSON Data Type: String Values accepted: <ul style="list-style-type: none">• 01 = Changed during this transaction• 02 = Less than 30 days• 03 = 30–60 days• 04 = More than 60 days
Cardholder Account Date Field name: chAccDate	Date converted into UTC that the Cardholder opened the account with the 3DS Requestor.	Length: 8 characters JSON Data Type: String Format accepted: <ul style="list-style-type: none">• Date format = YYYYMMDD
Cardholder Account Requestor ID Field Name: chAccReqID	The 3DS Requestor assigned account identifier of the transacting Cardholder. This identifier is coded as the SHA-256 + Base64url of the account identifier for the 3DS Requestor and is provided as a String.	Length: Maximum 64 characters JSON Data Type: String
Cardholder Account Password Change Field Name: chAccPwChange	Date converted into UTC that Cardholder's account with the 3DS Requestor had a password change or account reset.	Length: 8 characters JSON Data Type: String Format accepted: <ul style="list-style-type: none">• Date format = YYYYMMDD

Data Element/Field Name	Description	Length/Format/Values
Cardholder Account Password Change Indicator Field Name: chAccPwChangeInd	Indicates the length of time since the Cardholder's account with the 3DS Requestor had a password change or account reset.	Length: 2 characters JSON Data Type: String Values accepted: <ul style="list-style-type: none"> • 01 = No change • 02 = Changed during this transaction • 03 = Less than 30 days • 04 = 30–60 days • 05 = More than 60 days
Cardholder Account Purchase Count Field Name: nbPurchaseAccount	Number of purchases with this cardholder account during the previous six months. If the Cardholder Account Purchase Count reaches the value 999, it remains set at 999.	Length: Maximum 4 characters JSON Data Type: String Values accepted: <ul style="list-style-type: none"> • 0–999
Number of Provisioning Attempts Per Day Field Name: provisionAttemptsDay	Number of Add Card attempts in the last 24 hours. Example values: <ul style="list-style-type: none"> • 2 • 02 • 002 	Length: Maximum 3 characters JSON Data Type: String
Number of Transactions Per Day Field Name: txnActivityDay	Number of transactions (successful and abandoned) for this cardholder account with the 3DS Requestor across all payment accounts in the previous 24 hours. Example values: <ul style="list-style-type: none"> • 2 • 02 • 002 	Length: Maximum 3 characters JSON Data Type: String

Data Element/Field Name	Description	Length/Format/Values
Number of Transactions Per Year Field Name: txnActivityYear	Number of transactions (successful and abandoned) for this cardholder account with the 3DS Requestor across all payment accounts in the previous year. If the maximum value is reached, the Number of Transactions Per Year remains set at 999. Example values: <ul style="list-style-type: none"> • 2 • 02 • 002 	Length: Maximum 3 characters JSON Data Type: String Values accepted: <ul style="list-style-type: none"> • 0–999
Payment Account Age Field Name: paymentAccAge	Date converted into UTC that the payment account was enrolled in the Cardholder's account with the 3DS Requestor.	Length: 8 characters JSON Data Type: String Format accepted: <ul style="list-style-type: none"> • Date format = YYYYMMDD
Payment Account Age Indicator Field Name: paymentAccInd	Indicates the length of time that the payment account was enrolled in the Cardholder's account with the 3DS Requestor.	Length: 2 characters JSON Data Type: String Values accepted: <ul style="list-style-type: none"> • 01 = No account (guest checkout) • 02 = During this transaction • 03 = Less than 30 days • 04 = 30–60 days • 05 = More than 60 days

Data Element/Field Name	Description	Length/Format/Values
Shipping Address Usage Field Name: shipAddressUsage	Date converted into UTC when the shipping address used for this transaction was first used with the 3DS Requestor.	Length: 8 characters JSON Data Type: String Format accepted: <ul style="list-style-type: none"> • Date format = YYYYMMDD
Shipping Address Usage Indicator Field Name: shipAddressUsageInd	Indicates when the shipping address used for this transaction was first used with the 3DS Requestor.	Length: 2 characters JSON Data Type: String Values accepted: <ul style="list-style-type: none"> • 01 = This transaction • 02 = Less than 30 days • 03 = 30–60 days • 04 = More than 60 days
Shipping Name Indicator Field Name: shipNameIndicator	Indicates if the Cardholder Name on the account is identical to the shipping Name used for this transaction.	Length: 2 characters JSON Data Type: String Values accepted: <ul style="list-style-type: none"> • 01 = Account Name identical to shipping Name • 02 = Account Name different than shipping Name
Suspicious Account Activity Field Name: suspiciousAccActivity	Indicates whether the 3DS Requestor has experienced suspicious activity (including previous fraud) on the cardholder account.	Length: 2 characters JSON Data Type: String Values accepted: <ul style="list-style-type: none"> • 01 = No suspicious activity has been observed • 02 = Suspicious activity has been observed

A.13.2 Merchant Risk Indicator

The Merchant Risk Indicator contains optional information about the specific purchase by the Cardholder. The detailed data elements, which are optional are outlined in Table A.11.

Table A.11: Merchant Risk Indicator

Data Element/Field Name	Description	Length/Format/Values
Delivery Email Address Field Name: deliveryEmailAddress	For Electronic delivery, the email address to which the merchandise was delivered.	Length: maximum 254 characters JSON Data Type: String
Delivery Timeframe Field name: deliveryTimeframe	Indicates the merchandise delivery timeframe.	Length: 2 characters JSON Data Type: String Values accepted: <ul style="list-style-type: none">• 01 = Electronic Delivery• 02 = Same-day shipping• 03 = Overnight shipping• 04 = Two-day or more shipping
Gift Card Amount Field Name: giftCardAmount	For prepaid or gift card purchase, the purchase amount total of prepaid or gift card(s) in major units (for example, USD 123.45 is 123). Example: gift card amount is USD 123.45: Values accepted: <ul style="list-style-type: none">• 123• 0123• 00123	Length: maximum 15 characters JSON Data Type: String

Data Element/Field Name	Description	Length/Format/Values
Gift Card Count Field Name: giftCardCount	For prepaid or gift card purchase, total count of individual prepaid or gift cards/codes purchased.	Length: 2 characters JSON Data Type: String
Gift Card Currency Field Name: giftCardCurr	For prepaid or gift card purchase, ISO 4217 three-digit currency code of the gift card, other than those listed in Table A.5.	Length: 3 characters; numeric JSON Data Type: String
Pre-Order Date Field Name: preOrderDate	For a pre-ordered purchase, the expected date that the merchandise will be available.	Length: 8 characters JSON Data Type: String Format accepted: <ul style="list-style-type: none">• Date format = YYYYMMDD
Pre-Order Purchase Indicator Field Name: preOrderPurchaseInd	Indicates whether Cardholder is placing an order for merchandise with a future availability or release date.	Length: 2 characters JSON Data Type: String Values accepted: <ul style="list-style-type: none">• 01 = Merchandise available• 02 = Future availability
Reorder Items Indicator Field Name: reorderItemsInd	Indicates whether the cardholder is reordering previously purchased merchandise.	Length: 2 characters JSON Data Type: String Values accepted: <ul style="list-style-type: none">• 01 = First time ordered• 02 = Reordered

Data Element/Field Name	Description	Length/Format/Values
Shipping Indicator Field Name: shipIndicator	<p>Indicates shipping method chosen for the transaction.</p> <p>Merchants must choose the Shipping Indicator code that most accurately describes the Cardholder's specific transaction, not their general business.</p> <p>If one or more items are included in the sale, use the Shipping Indicator code for the physical goods, or if all digital goods, use the Shipping Indicator code that describes the most expensive item.</p>	<p>Length: 2 characters</p> <p>JSON Data Type: String</p> <p>Values accepted:</p> <ul style="list-style-type: none"> • 01 = Ship to Cardholder's billing address • 02 = Ship to another verified address on file with merchant • 03 = Ship to address that is different than the Cardholder's billing address • 04 = "Ship to Store" / Pick-up at local store (Store address shall be populated in shipping address fields) • 05 = Digital goods (includes online services, electronic gift cards and redemption codes) • 06 = Travel and event tickets, not shipped • 07 = Other (for example, gaming, digital services not shipped, emedia subscriptions, etc.) • 08 = Pick-up and go delivery • 09 = Locker delivery (or other automated pick-up)

A.13.3 3DS Requestor Authentication Information

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

Table A.12: 3DS Requestor Authentication Information

Data Element/Field Name	Description	Length/Format/Values
3DS Requestor Authentication Data Field Name: threeDSReqAuthData	<p>Data that documents and supports a specific authentication process. In the current version of the specification, this data element is not defined in detail. However, the intention is that, for each 3DS Requestor Authentication Method, this field carry data that the ACS can use to verify the authentication process.</p> <p>For example, if the 3DS Requestor Authentication Method is:</p> <ul style="list-style-type: none">• 03, then this element can carry information about the provider of the federated ID and related information.• 06, then this element can carry the FIDO Assertion and/or Attestation Data.• 07, then this element can carry FIDO Assertion and/or Attestation Data with the FIDO Assurance Data signed by a trusted third party.• 08, then this element can carry the SRC Assurance Data. <p>For 3DS Requestor Authentication Method = 06 or 07, refer to the <i>EMV® 3-D Secure White Paper – Use of FIDO® Data in 3-D Secure Messages</i> for the 3DS Requestor Authentication Data content and format.</p>	Length: Maximum 20000 characters JSON Data Type: String or Object

3DS Requestor Authentication Method Field Name: threeDSReqAuthMethod	Mechanism used by the Cardholder to authenticate to the 3DS Requestor. Note: For 09 = SPC Authentication, the Assertion Data is provided as a JSON object returned by the SPC API.	Length: 2 characters JSON Data Type: String Values accepted: <ul style="list-style-type: none">• 01 = No 3DS Requestor authentication occurred (i.e., cardholder “logged in” as guest)• 02 = Login to the cardholder account at the 3DS Requestor system using 3DS Requestor’s own credentials• 03 = Login to the cardholder account at the 3DS Requestor system using federated ID• 04 = Login to the cardholder account at the 3DS Requestor system using issuer credentials• 05 = Login to the cardholder account at the 3DS Requestor system using third-party authentication• 06 = Login to the cardholder account at the 3DS Requestor system using FIDO Authenticator• 07 = Login to the cardholder account at the 3DS Requestor system using FIDO Authenticator (FIDO Assertion or Attestation data signed)• 08 = SRC Assurance Data• 09 = SPC Authentication• 10–79 = Reserved for EMVCo future use (values invalid until defined by EMVCo)• 80–99 = Reserved for DS use
--	---	---

Data Element/Field Name	Description	Length/Format/Values
3DS Requestor Authentication Timestamp Field name: threeDSReqAuthTimestamp	Date and time of the cardholder authentication converted into UTC.	Length: 12 characters JSON Data Type: String Format accepted: <ul style="list-style-type: none">• Date format = YYYYMMDDHHMM

A.13.4 3DS Requestor Prior Transaction Authentication Information

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

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

Data Element/Field Name	Description	Length/Format/Values
3DS Requestor Prior DS Transaction ID Field Name: threeDSReqPriorDsTransId	This data element provides the prior DS Transaction ID to the ACS to determine the best approach for handling a request.	Length: 36 characters JSON Data Type: String Value accepted: <ul style="list-style-type: none">• This data element contains a DS Transaction ID for a prior authenticated transaction (for example, the first recurring transaction that was authenticated with the Cardholder).

Data Element/Field Name	Description	Length/Format/Values
3DS Requestor Prior Transaction Authentication Data Field Name: threeDSReqPriorAuthData	Data that documents and supports a specific authentication process. In the current version of the specification this data element is not defined in detail. However, the intention is that for each 3DS Requestor Authentication Method, this field carry data that the ACS can use to verify the authentication process. In future versions of the specification, these details are expected to be included.	Length: maximum 20000 characters JSON Data Type: String
3DS Requestor Prior Transaction Authentication Method Field Name: threeDSReqPriorAuthMethod	Mechanism used by the Cardholder to previously authenticate to the 3DS Requestor.	Length: 2 characters JSON Data Type: String Values accepted: <ul style="list-style-type: none"> • 01 = Frictionless authentication occurred by ACS • 02 = Cardholder challenge occurred by ACS • 03 = AVS verified • 04 = Other issuer methods • 05 = SPC authentication • 06–79 = Reserved for EMVCo future use (values invalid until defined by EMVCo) • 80–99 = Reserved for DS use
3DS Requestor Prior Transaction Authentication Timestamp Field name: threeDSReqPriorAuthTimestamp	Date and time converted into UTC of the prior Cardholder authentication.	Length: 12 characters JSON Data Type: String Format accepted: <ul style="list-style-type: none"> • Date format = YYYYMMDDHHMM

Data Element/Field Name	Description	Length/Format/Values
3DS Requestor Prior Transaction Reference Field Name: threeDSReqPriorRef	This data element provides additional information to the ACS to determine the best approach for handing a request.	Length: 36 characters JSON Data Type: String Value accepted: <ul style="list-style-type: none">• This data element contains an ACS Transaction ID for a prior authenticated transaction (for example, the first recurring transaction that was authenticated with the cardholder).

A.13.5 ACS Rendering Type

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

Table A.14: ACS Rendering Type

Data Element/Field Name	Description	Length/Format/Values	Inclusion
ACS Interface Field Name: acsInterface	This the ACS interface that the challenge will present to the Cardholder.	Length: 2 characters JSON Data Type: String Values accepted: <ul style="list-style-type: none">• 01 = Native UI• 02 = HTML UI	R

Data Element/Field Name	Description	Length/Format/Values	Inclusion
ACS UI Template Field Name: acsUiTemplate	Identifies the UI Template format that the ACS first presents to the Cardholder. Valid values for each Interface: <ul style="list-style-type: none"> • Native UI = 01–04, 07 • HTML UI = 01–07 Note: HTML Other and HTML OOB are only valid in combination with 02 = HTML UI. If used with 01 = Native UI, the DS will respond with Error = 203 as described in sections 5.9.3 and 5.9.8.	Length: 2 characters JSON Data Type: String Values accepted: <ul style="list-style-type: none"> • 01 = Text • 02 = Single Select • 03 = Multi Select • 04 = OOB • 05 = HTML Other • 06 = HTML OOB • 07 = Information 	R
Device User Interface Mode Field Name: deviceUserInterfaceMode	Indicates the user interface mode the ACS will present to the Cardholder for a challenge.	Length: 2 numeric characters JSON Data Type: String Values accepted: <ul style="list-style-type: none"> • 01 = Portrait • 02 = Landscape • 03 = Voice • 04 = Other 	R

JSON Object Example

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

A.13.6 Device Rendering Options Supported

The Device Rendering Options Supported contains information about the rendering types and interface that the device supports. The detailed data elements are outlined in Table A.15.

Note: All Device Rendering Options must be supported by all components.

Table A.15: Device Rendering Options Supported

Data Element/Field Name	Description	Length/Format/Values	Inclusion
SDK Authentication Type Field Name: sdkAuthenticationType	Authentication methods preferred by the 3DS SDK in order of preference.	Size: 1–99 elements JSON Data Type: Array of String String: 2 characters Values accepted: <ul style="list-style-type: none">• 01 = Static Passcode• 02 = SMS OTP• 03 = Key fob or EMV card reader OTP• 04 = App OTP• 05 = OTP Other• 06 = KBA• 07 = OOB Biometrics• 08 = OOB Login• 09 = OOB Other• 10 = Other• 11 = Push Confirmation• 12–79 = Reserved for future EMVCo use (values invalid until defined by EMVCo)• 80–99 = Reserved for DS use	O

Data Element/Field Name	Description	Length/Format/Values	Inclusion
SDK Interface Field Name: sdkInterface	<p>Lists all of the SDK Interface types that the device supports for displaying specific challenge user interfaces within the 3DS SDK.</p>	<p>Length: 2 characters JSON Data Type: String Values accepted:</p> <ul style="list-style-type: none"> • 01 = Native • 02 = HTML • 03 = Both 	R
SDK UI Type Field Name: sdkUiType	<p>Lists all UI types that the device supports for displaying specific challenge user interfaces within the 3DS SDK.</p> <p>Valid values for each Interface:</p> <ul style="list-style-type: none"> • Native UI = 01–04, 07 • HTML UI = 01–07 <p>Note: Currently, all 3DS SDKs need to support all UI Types. In the future, however, this may change (for example, smart watches may support a UI Type not yet defined by this specification).</p>	<p>Size: Variable, 1–7 elements JSON Data Type: Array of String. String: 2 characters Values accepted:</p> <ul style="list-style-type: none"> • 01 = Text • 02 = Single Select • 03 = Multi Select • 04 = OOB • 05 = HTML Other (valid only for HTML UI) • 06 = HTML OOB (valid only for HTML UI) • 07 = Information 	R

JSON Object Example:

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

A.13.7 Challenge Data Entry

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

Table A.16: Challenge Data Entry

Challenge Data Entry	ACS UI Type	Challenge Cancelation Indicator	Resend Challenge Information Code	Challenge Additional Code	Challenge No Entry	Response
Missing	01, 02, or 03	Missing	Missing	Missing	Present <ul style="list-style-type: none"> • Value = Y 	The ACS assumes that the Cardholder has not entered challenge data in the UI and therefore the ACS does not send the 3DS SDK an Error Message, but instead sends a CRes message.
Missing	01, 02, or 03	Present	Missing	Missing	Missing	The ACS sends the 3DS SDK a CRes message.
Missing	01, 02, or 03	Missing	Present <ul style="list-style-type: none"> • Value = Y 	Missing	Missing	The ACS sends the 3DS SDK a CRes message.
Missing	01, 02, or 03	Missing	Missing	Present <ul style="list-style-type: none"> • Value = Y 	Missing	The ACS sends the 3DS SDK a CRes message.
Missing	01, 02, or 03	Missing	Missing	Present <ul style="list-style-type: none"> • Value = N 	Present <ul style="list-style-type: none"> • Value = Y 	The ACS assumes that the Cardholder has not entered challenge data in the UI and therefore the ACS does not send the 3DS SDK an Error Message, but instead sends a CRes message.

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

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

A.13.8 Transaction Status Conditions

The Transaction Status (*transStatus*) indicates whether a transaction qualifies as an authenticated transaction or account verification. The conditions on which indicators are valid within the 3-D Secure messages are outlined in Table A.17.

Table A.17: Transaction Status Conditions

Transaction Status	ARes	Final CRes	RReq	Error Response
Y = Authentication Verification Successful	Valid	Valid	Valid	Not Applicable
N = Not Authenticated/Account Not Verified; Transaction denied	Valid	Valid	Valid	Not Applicable
U = Authentication/Account Verification Could Not Be Performed; Technical or other problem, as indicated in the ARes or RReq	Valid	Invalid	Valid	Final CRes: End processing (no Error)
A = Attempts Processing Performed; Not Authenticated/Verified, but a proof of attempted authentication/verification is provided	Valid	Invalid	Valid	Final CRes: End processing (no Error)
C = Challenge Required; Additional authentication is required using the CReq/CRes	Valid ⁹	Invalid	Invalid	<ul style="list-style-type: none">• ARes: Refer to Section 5.9.3 and use Error Code = 203 if Condition not met• Final CRes: End processing (no Error)

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

Transaction Status	ARes	Final CRes	RReq	Error Response
D = Challenge Required; Decoupled Authentication confirmed	Valid ¹⁰	Invalid	Valid ¹¹	<ul style="list-style-type: none"> • ARes: Refer to Section 5.9.3 and use Error Code = 203 if Condition not met • Final CRes: End processing (no Error)
R = Authentication/ Account Verification Rejected; Issuer is rejecting authentication/verification and request that authorisation not be attempted	Valid	Invalid	Valid	Final CRes: End processing (no Error)
I = Informational Only; 3DS Requestor challenge preference acknowledged	Valid ¹²	Invalid	Invalid	<ul style="list-style-type: none"> • ARes: Refer to Section 5.9.3 and use Error Code = 203 if Condition not met • Final CRes: End processing (no Error) • RReq: Refer to Section 5.9.8 and use Error Code = 203
S = Challenge using SPC	Valid ¹³	Invalid	Invalid	<ul style="list-style-type: none"> • ARes: Refer to Section 5.9.3 and use Error Code = 203 if Condition not met • Final CRes: End processing (no Error) • RReq: Refer to Section 5.9.8 and use Error Code = 203

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

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

¹² This indicator (I) can be sent only if 3DS Requestor Challenge Indicator = 05, 06, or 07 within the AReq message (or as specified by DS rules).

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

A.13.9 Multi-Transaction

The Multi-Transaction object contains optional information about a specific purchase by the Cardholder invoking multiple transactions or merchants. The detailed data elements are outlined in Table A.18.

Table A.18: Multi-Transaction

Data Element/Field Name	Description	Length/Format/Values	Inclusion
Merchant List Field Name: merchantList	<p>Contains the details of each merchant involved in the transaction.</p> <p>JSON array of objects containing:</p> <ul style="list-style-type: none"> • Merchant Name Listed • Acquirer Merchant ID Listed • Merchant Amount • Merchant Currency Code • Merchant Currency Exponent • Seller ID 	<p>Size: Variable, maximum 1–50 elements</p> <p>JSON Data Type: Array of Objects</p> <p>Values accepted:</p> <ul style="list-style-type: none"> • Refer to the following data elements: <ul style="list-style-type: none"> ◦ Merchant Name Listed ◦ Acquirer Merchant ID Listed ◦ Merchant Amount ◦ Merchant Currency Code ◦ Merchant Currency Exponent ◦ Seller ID 	Required
Merchant Name Listed Field name: merchantNameListed	Name of the listed merchant	<p>Length: Variable, maximum 40 characters</p> <p>JSON Data Type: String</p> <p>Value accepted:</p> <ul style="list-style-type: none"> • Name used in the authorisation message as defined in ISO 8583-1. 	Required
Acquirer Merchant ID Listed Field Name: acquirerMerchantIdListed	<p>Acquirer-assigned Merchant Listed Identifier.</p> <p>This may be the same value that is used in authorisation requests sent on behalf of the 3DS Requestor and is represented in ISO 8583-1 formatting requirements.</p>	<p>Length: Variable, maximum 15 characters</p> <p>JSON Data Type: String</p>	Optional

Data Element/Field Name	Description	Length/Format/Values	Inclusion
Merchant Amount Field Name: merchantAmount	Purchase amount for the merchant in minor units of currency with all punctuation removed. When used in conjunction with the Purchase Currency Exponent field, proper punctuation can be calculated.	Length: Variable, maximum 48 characters JSON Data Type: String	Optional
Merchant Currency Code Field Name: merchantCurrency	Currency Code in which purchase Merchant Amount is expressed. ISO 4217 three-digit currency code, other than those listed in Table A.5.	Length: 3 characters; numeric JSON Data Type: String	Required if Merchant Amount is present
Merchant Currency Exponent Field Name: merchantExponent	Minor units of Merchant Currency as specified in the ISO 4217 currency exponent.	Length: 1 character JSON Data Type: String	Required if Merchant Amount is present
Seller ID Field Name: sellerId	Merchant-assigned Seller identifier that links additional Seller Information outlined in Table A.19. Note: If this data element is present, this must match the Seller ID field in the Seller Information object.	Length: Variable, maximum 50 characters JSON Data Type: String	Optional
AV Validity Time Field Name: avValidityTime	Number of days that the AV (Authentication Value) is valid.	Length: 1–3 characters; numeric JSON Data Type: String Values accepted: <ul style="list-style-type: none">• 0–999	Conditional based on DS rules

Data Element/Field Name	Description	Length/Format/Values	Inclusion
AV Number Use Field Name: avNumberUse	Number of times that the AV (Authentication Value) is valid.	Length: 1–2 characters; numeric JSON Data Type: String Values accepted: <ul style="list-style-type: none">• 0–99	Conditional based on DS rules

JSON Object Example

```
{  
    "merchantList": [  
        {  
            "merchantNameListed": "Merchant 1",  
            "acquirerMerchantIdListed": "Acquirer Merchant Id Merchant 1",  
            "merchantAmount": "100",  
            "merchantCurrency": "840",  
            "merchantExponent": "2",  
            "sellerId": "Seller Id 1"  
        },  
        {  
            "merchantNameListed": "Merchant 2",  
            "acquirerMerchantIdListed": "Acquirer Merchant Id Merchant 2",  
            "merchantAmount": "20000",  
            "merchantCurrency": "392",  
            "merchantExponent": "0",  
            "sellerId": "Seller Id 2",  
        },  
        ],  
        "avValidityTime": "2",  
        "avNumberUse": "3"  
    }
```

A.13.10 Seller Information

The Seller Information object contains optional information about a specific purchase by the Cardholder invoking transactions where merchants submit transaction details on behalf of another entity, i.e., individual sellers in a marketplace or drivers in a ride share platform. The detailed data elements are outlined in Table A.19.

Table A.19: Seller Information

Data Element/Field Name	Description	Length/Format/Values	Inclusion
Seller Information Field Name: sellerInfo	<p>Contains the details of each seller involved in the transaction.</p> <p>JSON array of objects containing:</p> <ul style="list-style-type: none"> • Seller Name • Seller ID • Seller Business Name • Seller Account Date • Seller Address Line 1 • Seller Address Line 2 • Seller Address Line 3 • Seller Address City • Seller Address State • Seller Address Postal Code • Seller Address Country • Seller Email Address • Seller Phone Number 	<p>Size: Variable, 1–50 elements</p> <p>JSON Data Type: Array of Objects</p> <p>Values accepted:</p> <ul style="list-style-type: none"> • Refer to the following data elements: <ul style="list-style-type: none"> ◦ Seller Name ◦ Seller ID ◦ Seller Business Name ◦ Seller Account Date ◦ Seller Address Line 1 ◦ Seller Address Line 2 ◦ Seller Address Line 3 ◦ Seller Address City ◦ Seller Address State ◦ Seller Address Postal Code ◦ Seller Address Country ◦ Seller Email Address ◦ Seller Phone Number 	Required
Seller Name Field name: sellerName	Name of the Seller.	<p>Length: Variable, maximum 100 characters</p> <p>JSON Data Type: String</p>	Required

Data Element/Field Name	Description	Length/Format/Values	Inclusion
Seller ID Field Name: sellerId	Merchant-assigned Seller identifier.	Length: Variable, maximum 50 characters JSON Data Type: String	Required if Seller ID in Multi-Transaction object is present.
Seller Business Name Field Name: sellerBusinessName	Business name of the Seller.	Length: Variable, maximum 100 characters JSON Data Type: String	Optional
Seller Account Date Field Name: sellerAccDate	Date converted into UTC that the Seller started using the Merchant's services.	Length: 8 characters JSON Data Type: String Format accepted: <ul style="list-style-type: none">• Date format = YYYYMMDD	Optional
Seller Address Line 1 Field Name: sellerAddrLine1	First line of the business or contact street address of the Seller.	Length: Variable, maximum 50 characters JSON Data Type: String	Optional
Seller Address Line 2 Field Name: sellerAddrLine2	Second line of the business or contact street address of the Seller.	Length: Variable, maximum 50 characters JSON Data Type: String	Optional
Seller Address Line 3 Field Name: sellerAddrLine3	Third line of the business or contact street address of the Seller.	Length: Variable, maximum 50 characters JSON Data Type: String	Optional
Seller Address City Field Name: sellerAddrCity	Business or contact city of the Seller.	Length: Variable, maximum 50 characters JSON Data Type: String	Optional

Data Element/Field Name	Description	Length/Format/Values	Inclusion
Seller Address State Field Name: sellerAddrState	Business or contact state or province of the Seller.	Length: Variable, maximum 3 characters JSON Data Type: String Value accepted: <ul style="list-style-type: none">• Country subdivision code defined in ISO 3166-2. For example, using the ISO entry US-CA (California, United States), the correct value for this field = CA. Note that the country and hyphen are not included in this value.	Optional
Seller Address Postal Code Field Name: sellerAddrPostCode	Business or contact ZIP or other postal code of the Seller.	Length: Variable, maximum 16 characters JSON Data Type: String	Optional
Seller Address Country Field Name: sellerAddrCountry	Business or contact country of the Seller.	Length: 3 characters JSON Data Type: String Values accepted: <ul style="list-style-type: none">• ISO 3166-1 numeric three-digit country code, other than exceptions listed in Table A.5.	Optional
Seller Email Address Field Name: sellerEmail	Business or contact email address of the Seller.	Length: Variable, maximum 254 characters JSON Data Type: String Values accepted: <ul style="list-style-type: none">• Shall meet requirements of Section 3.4 of IETF RFC 5322.	Optional

Data Element/Field Name	Description	Length/Format/Values	Inclusion
Seller Phone Number Field Name: sellerPhone	Business or contact phone number of the Seller.	Length: Variable <ul style="list-style-type: none">• cc: 1-3 characters• subscriber: variable, maximum 15 characters Format: JSON Object; strings Values accepted: <ul style="list-style-type: none">• Country Code and Subscriber sections of the number represented by the following named fields:<ul style="list-style-type: none">◦ cc◦ subscriber Refer to ITU-E.164 for additional information on format and length. Example: <pre>"sellerPhone": { "cc": "1", "subscriber": "1234567899" }</pre>	Optional

JSON Object Example

```
{  
    "sellerInfo": [  
        {  
            "sellerName": "Seller 1",  
            "sellerId": "Seller Id 1",  
            "sellerPhone": {  
                "cc": "1",  
                "subscriber": "1234567899"  
            }  
        }  
    ]  
}
```

```
"sellerBusinessName": "Seller Business 1",
"sellerAccDate": "20210928",
"sellerAddrLine1": "Seller1 avenue 101",
"sellerAddrCity": "Seller1 city",
"sellerAddrState": "AZ",
"sellerAddrPostCode": "43121",
"sellerAddrCountry": "840",
"sellerEmail": "seller1@seller1.com"
},
{
  "sellerName": "Seller 2",
  "sellerId": "Seller Id 2",
  "sellerBusinessName": "Seller Business 2",
  "sellerAccDate": "20201022",
  "sellerAddrLine1": "Seller2 road",
  "sellerAddrCity": "Seller2 city",
  "sellerAddrState": "CA",
  "sellerAddrPostCode": "94212",
  "sellerAddrCountry": "840",
  "sellerPhone": {"cc": "1", "subscriber": "1234567899"}
}
]
```

A.14 UI Data Elements

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

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

Table A.20: UI Data Elements

Data Element / Field Name	Zone	Display Order (Top-down)		ACS UI Type				
		Portrait	Landscape	01 = Text	02 = Single Select	03 = Multi Select	04 = OOB	07 = Information
Challenge Additional Label Field Name: challengeAddLabel	3	11	9	O	O	O	O	O
Challenge Entry Box Field Name: challengeEntryBox	3	5	5	M	N	N	N	N
Challenge Entry Box 2 Field Name: challengeEntryBoxTwo	3	6	5 or 6	O	N	N	N	N
Challenge Information Header Field Name: challengeInfoHeader	3	2	2	M	M	M	M	M

Data Element / Field Name	Zone	Display Order (Top-down)		ACS UI Type				
		Portrait	Landscape	01 = Text	02 = Single Select	03 = Multi Select	04 = OOB	07 = Information
Challenge Data Entry Masking Toggle Field Name: challengeDataEntryToggle	3	5	5	O	N	N	N	N
Challenge Information Label Field Name: challengeInfoLabel	3	4	4	N	M	M	O	O
Challenge Information Text Field Name: challengeInfoText	3	3	3	M	M	M	M	M
Challenge Information Text Indicator Field Name: challengeInfoTextIndicator	3	3	3	O	O	O	O	O
Challenge Selection Information Field Name: challengeSelectInfo	3	5	5	N	M	M	N	N
Device Binding Information Text ¹⁴ Field Name: deviceBindingInfoText	4	8 or 13	8 or 12	O	O	O	O	O
Expandable Information Label Field Name: expandInfoLabel	4	16	12	O	O	O	O	O

¹⁴ Device Binding Information Text display order depends on the Toggle Position Indicator.

Data Element / Field Name	Zone	Display Order (Top-down)		ACS UI Type				
		Portrait	Landscape	01 = Text	02 = Single Select	03 = Multi Select	04 = OOB	07 = Information
Expandable Information Text Field Name: expandInfoText	4	17	13	O	O	O	O	O
Information Continuation Label Field Name: infoContinueLabel	3	11	9	N	N	N	N	M
Issuer Image ¹⁵ Field Name: issuerImage	2	1	1	C	C	C	C	C
OOB App Label ¹⁶ Field Name: oobAppLabel	3	10	9	N	N	N	C	N
OOB Continuation Label Field Name: oobContinueLabel	3	11	9	N	N	N	C	N
Payment System Image ¹⁷ Field Name: psImage	2	1	1	C	C	C	C	C

¹⁵ Refer to Table A.1 for Inclusion conditions.

¹⁶ Refer to Table A.1 for inclusion conditions.

¹⁷ Refer to Table A.1 for inclusion conditions.

Data Element / Field Name	Zone	Display Order (Top-down)		ACS UI Type				
		Portrait	Landscape	01 = Text	02 = Single Select	03 = Multi Select	04 = OOB	07 = Information
Resend Information Label Field Name: resendInformationLabel	3	11	9	O	N	N	N	N
Submit Authentication Label Field Name: submitAuthenticationLabel	3	10	9	M	M	M	N	N
Trust List Information Text ¹⁸ Field Name: trustListInfoText	3	8 or 13	7 or 10	O	O	O	O	O
Why Information Label Field Name: whyInfoLabel	4	15	10	O	O	O	O	O
Why Information Text Field Name: whyInfoText	4	16	11	O	O	O	O	O

¹⁸ Trust List Information Text display order depends on the Toggle Position Indicator

A.14.1 Issuer Image

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

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

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

Table A.21: Issuer Image

Data Element/Field Name	Description	Length/Format/Values
<ul style="list-style-type: none">• Default Image Field Name: <code>default</code>• Dark Mode Image Field Name: <code>dark</code>• Monochrome Image Field Name: <code>monochrome</code>	<p>Include at minimum one and at maximum three Fully Qualified URLs defined as default, dark mode or monochrome images of the Issuer Image.</p> <p>Examples:</p> <pre>"issuerImage" : { "default": "https://acs.com/default_image.png", "dark": "https://acs.com/dark_image.png", "monochrome": "https://acs.com/monochrome_image.png" }</pre>	<p>Length: Variable, maximum 6144 characters</p> <p>JSON Data Type: JSON object</p> <p>Value accepted:</p> <ul style="list-style-type: none">• Fully Qualified URL in correct JSON Object format <p>If present, the Issuer Image object shall contain, at minimum, the Default Image.</p>

A.14.2 Payment System Image

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

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

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

Table A.22: Payment System Image

Data Element/Field Name	Description	Length/Format/Values
<ul style="list-style-type: none">• Default Image Field Name: <code>default</code>• Dark Mode Image Field Name: <code>dark</code>• Monochrome Image Field Name: <code>monochrome</code>	<p>Include at minimum one and at maximum three Fully Qualified URLs defined as default, dark mode or monochrome images of the DS or Payment System Image.</p> <p>Examples:</p> <pre>"psImage" : { "default": "https://ds.com/default_image.png", "dark": "https://ds.com/dark_image.png", "monochrome": "https://ds.com/monochrome_image.png" }</pre>	<p>Length: Variable, maximum 6144 characters JSON Data Type: JSON object Value accepted:</p> <ul style="list-style-type: none">• Fully Qualified URL in correct JSON Object format <p>If present, the Payment System Image object shall contain, at minimum, the Default Image.</p>

A.15 iframe and Sandbox Attributes

Table A.23 specifies the iframe attributes that the 3DS Requestor uses when it creates the challenge or 3DS Method iframe.

Table A.23: iframe Attributes

Attribute ¹⁹	Value
allowfullscreen	false
allowpaymentrequest	false
height	as per Challenge Window Size
sandbox	refer to Table A.24
srcdoc	may be used to initialise redirection content
width	as per Challenge Window Size
allow="payment *; publickey-credentials-get *" ²⁰	enable access to WebAuthn and SPC (Secure Payment Confirmation) API and Payment Request API

¹⁹ Attributes not listed in Table A.23 should not be present.

²⁰ Use the following syntax <iframe src="<https://www.foo.com>" allow="payment *; publickey-credentials-get *"></iframe>, if supported by the Browser.

Table A.24 specifies the sandbox attributes that the 3DS Requestor uses when it creates the challenge or 3DS Method iframe.

Table A.24: Sandbox Attributes

Attribute	Description	Inclusion
allow-forms	Allows the processing of forms.	R
allow-scripts	Allows the processing of scripts. Note the <code>allow-scripts</code> permission does not give the iframe the ability to create pop-ups or modal windows, which can help prevent clickjacking attacks from occurring.	R
allow-same-origin	Gives the iframe permission to only use the data from the same ACS domain.	R
allow-pointer-lock	Gives access to the mouse position and events. Note: this attribute is not needed for the 3DS Method iframe.	R
allow-downloads-without-user-activation	Prevents downloads to be initiated for content in the iframe without user action.	Not Allowed
allow-downloads	Prevents downloads to be initiated for content in the iframe.	Not Allowed
allow-modals	Prevents to open modal window from the iframe.	Not Allowed
allow-orientation-lock	Prevents to lock the screen orientation.	Not Allowed
allow-popups	Prevents popup windows.	Not Allowed
allow-popups-to-escape-sandbox	Prevents popups to open new windows without inheriting the sandboxing.	Not Allowed
allow-presentation	Prevents to initiate a presentation session.	Not Allowed
allow-storage-access-by-user-activation	Prevents access to the parent's storage capabilities.	Not Allowed

Attribute	Description	Inclusion
allow-top-navigation	Prevents access to the top-level browsing context.	Not Allowed
allow-top-navigation-by-user-activation	Prevents access to the top-level browsing context also with user interaction.	Not Allowed

A.16 3-D Secure Array Fields

A 3-D Secure array is defined to contain either:

- string
- array
- object

A 3-D Secure array is not defined to contain either:

- number
- boolean
- null
- duplicate elements (i.e., identical string, object or array)

Duplicate elements are not allowed in an array (i.e., identical string, object or array). For example: [{"phone": "Mobile **** * 321"}, {"phone": "Mobile **** * 321"}]

In the case of duplicate elements in an array, the receiving component returns an Error Message as defined in Section A.9 with the applicable Error Component and Error Code = 204.

A.17 EMV Payment Token Information

Table A.25 specifies the EMV Payment Token Information that 3-D Secure components can provide or receive token-related information after a Payment Token has been de-tokenised.

Table A.25: Token Information

Data Element/Attribute Name	Description	Source	Length/Format/Values	Inclusion
Payment Token Attribute Name: <code>token</code>	Payment token used to initiate the EMV 3DS transaction.	3DS Server DS	Length: Variable, 13-19 characters JSON Data Type: String Value accepted: <ul style="list-style-type: none">• Format represented ISO 7812	AReq = O
Token Additional Data Attribute Name: <code>tokenAdditionalData</code>	Additional information about the Payment Token from the Token Service Provider.	3DS Server DS	Length: Variable, maximum 500 characters JSON Data Type: Object	AReq = O
Token Assurance Method Attribute Name: <code>tokenAssuranceMethod</code>	An updatable value that allows the Token Service Provider to communicate the ID&V performed. It is determined or updated by the ID&V Method(s) and ID&V Actor.	DS	Length: 2 characters JSON Data Type: String Values accepted: <ul style="list-style-type: none">• Refer to EMV Tokenisation Technical Framework.	AReq = O
Token Requestor ID Attribute Name: <code>tokenRequestorId</code>	An 11-digit numeric value that identifies each unique combination of Token Requestor and Token Domain(s) for a given Token Service Provider.	DS	Length: 11 characters JSON Data Type: String <ul style="list-style-type: none">• Refer to EMV Tokenisation Technical Framework.	AReq = O

Data Element/Attribute Name	Description	Source	Length/Format/Values	Inclusion
Token Cryptogram Attribute Name: tokenCryptogram	A cryptogram, containing a transaction-unique value, typically generated using the Payment Token, Payment Token related data and transaction data. Cryptogram derivation methods may vary by scenario and may be Payment System-specific.	3DS Server	Length: Variable, maximum 4000 characters JSON Data Type: String <ul style="list-style-type: none">• Refer to EMV Tokenisation Technical Framework.	AReq = O
Token Cryptogram Validity Indicator Attribute Name: tokenCryptogramValidityIndicator	Identifies if the Token Cryptogram has been verified and the outcome of that verification.	DS	Length: 2 characters JSON Data Type: String Values accepted: <ul style="list-style-type: none">• 01 = Verified• 02 = Failed• 03 = Not Performed• 04–79 = Reserved for EMVCo future use (values invalid until defined by EMVCo)• 80–99 = Reserved for DS use <p>Note: If the element is not provided, the expected action is for the ACS to interpret as 03.</p>	AReq = O
Token Status Indicator Attribute Name: tokenStatusIndicator	Identifies the current status of the Payment Token.	3DS Server DS	Length: Variable, maximum 40 characters JSON Data Type: String	AReq = O

A.18 Challenge Text Box Settings

Table A.26 specifies the text box settings that the ACS can use during a challenge for ACS UI Type = 01.

Table A.26: Text Box Settings

Data Element/Field Name	Description	Length/Format/Values	Inclusion
Challenge Data Entry Keyboard Type Field Name: challengeDataEntryKeyboardType	Indicates if the 3DS SDK shall display a numeric or alphanumeric keyboard.	Length: 2 characters JSON Data Type: String Values accepted: <ul style="list-style-type: none">• 01 = Numeric keyboard• 02 = Alphanumeric keyboard• 03–99 = Reserved for EMVCo future use (values invalid until defined by EMVCo)	O
Challenge Data Entry Autofill Field Name: challengeDataEntryAutofill	Indicates if the 3DS SDK enables the autofill option for the Challenge Data Entry. When enabled, the 3DS SDK/OS automatically copies the received or saved code or password in the Challenge Data Entry. If Challenge Data Entry Autofill is not present, the option is not enabled.	Length: 1 character JSON Data Type: String Values accepted: <ul style="list-style-type: none">• Y = Autofill supported for the Challenge Data Entry	O
Challenge Data Entry Autofill Type Field Name: challengeDataEntryAutofillType	Indicates the type of data expected when the Challenge Data Entry Autofill is active. Refer to the following for Android or iOS https://developer.android.com/reference/androidx/autofill/HintConstants https://developer.apple.com/documentation/security/password_autofill/enabling_password_autofill_on_a_text_input_view?language=objc	Length: 2 characters JSON Data Type: String Values accepted: <ul style="list-style-type: none">• 01 = SMS OTP• 02 = Password• 03–99 = Reserved for EMVCo future use (values invalid until defined by EMVCo)	C Required if Challenge Data Entry Autofill = Y
Challenge Data Entry Length Maximum Field Name: challengeDataEntryLengthMax	Indicates to the 3DS SDK the maximum length of the challenge data entry. Set default to 45 if not present.	Length: 2 characters JSON Data Type: String Values accepted: <ul style="list-style-type: none">• 01–45	O

Data Element/Field Name	Description	Length/Format/Values	Inclusion
Challenge Data Entry Label Field Name: challengeDataEntryLabel	Label to specify the expected data entry provided by the ACS.	Length: Variable, maximum 45 characters JSON Data Type: String	R
Challenge Data Entry Masking Field Name: challengeDataEntryMasking	Indicates that the 3DS SDK shall mask the data entered by the Cardholder.	Length: 1 character JSON Data Type: String Values accepted: <ul style="list-style-type: none">• Y = Mask the data entered by the Cardholder• N = Do not mask the data entered by the Cardholder.	O
Challenge Data Entry Masking Toggle Field Name: challengeDataEntryToggle	Indicates that the 3DS SDK shall display a toggle icon, and display the data entered, if selected by the Cardholder.	Length: 1 character JSON Data Type: String Values accepted: <ul style="list-style-type: none">• Y = Display the toggle icon indicator• N = Do not display the toggle icon indicator	Required when Challenge Data Entry Masking = Y

A.19 Broadcast Information

Table A.27 specifies the Broadcast Information object sent between the 3DS Server, the DS and the ACS.

Table A.27: Broadcast Information

Data Element/Field Name	Description	Source	Length/Format/Values	Inclusion
Category Field Name: category	Indicates the category/type of information.	3DS Server DS ACS	Length: 2 characters JSON Data Type: String Values accepted: <ul style="list-style-type: none">• 01 = General• 02 = Certificate expiry• 03 = Fraud alert• 04 = Operational alert• 05 = Transactional data• 06 = Other• 07–79 = Reserved for EMVCo future use (values invalid until defined by EMVCo)• 80–99 = Reserved for DS use	AReq = R ARes = R
Description Field Name: description	Information to be broadcasted to recipients.	3DS Server DS ACS	Length: Variable, maximum 4000 characters JSON Data Type: String	AReq = O ARes = O
Expiry Date Field Name: expDate	The date after which the relevance of the broadcasted information (e.g., certificate expiration dates) expires.	3DS Server DS ACS	Length: 8 characters JSON Data Type: String Format accepted: <ul style="list-style-type: none">• YYYYMMDD	AReq = O ARes = O

Data Element/Field Name	Description	Source	Length/Format/Values	Inclusion
Severity Field Name: severity	<p>Indicates the importance/severity level of the broadcasted information.</p> <ul style="list-style-type: none"> • Critical = Immediate action to be taken by recipient • Major = Major impact; Upcoming action to be taken by recipient • Minor = Minor impact; Upcoming action to be taken by recipient • Informational = Informational only with no immediate action by recipient 	3DS Server DS ACS	<p>Length: 2 characters JSON Data Type: String Values accepted:</p> <ul style="list-style-type: none"> • 01 = Critical • 02 = Major • 03 = Minor • 04 = Informational 	AReq = R ARes = R
Recipient(s) Field Name: recipients	Indicates the intended recipient(s) of the broadcasted information.	3DS Server DS ACS	<p>Size: Variable, maximum 3 elements JSON Data Type: Array of String String: 2 characters Values accepted:</p> <ul style="list-style-type: none"> • 01 = 3DS SDK • 02 = 3DS Server • 03 = DS • 04 = ACS 	AReq = R ARes = R
Source Field Name: source	Indicates the source of the broadcasted information.	3DS Server DS ACS	<p>Size: 2 characters JSON Data Type: String Values accepted:</p> <ul style="list-style-type: none"> • 01 = 3DS Server • 02 = DS • 03 = ACS 	AReq = R ARes = R

A.20 Cardholder Information Text

Figure A.1 and Figure A.2 provide a sample UI format to display the Cardholder Information Text to the Cardholder.

Figure A.1 Sample Cardholder Information UI Example—App—Portrait

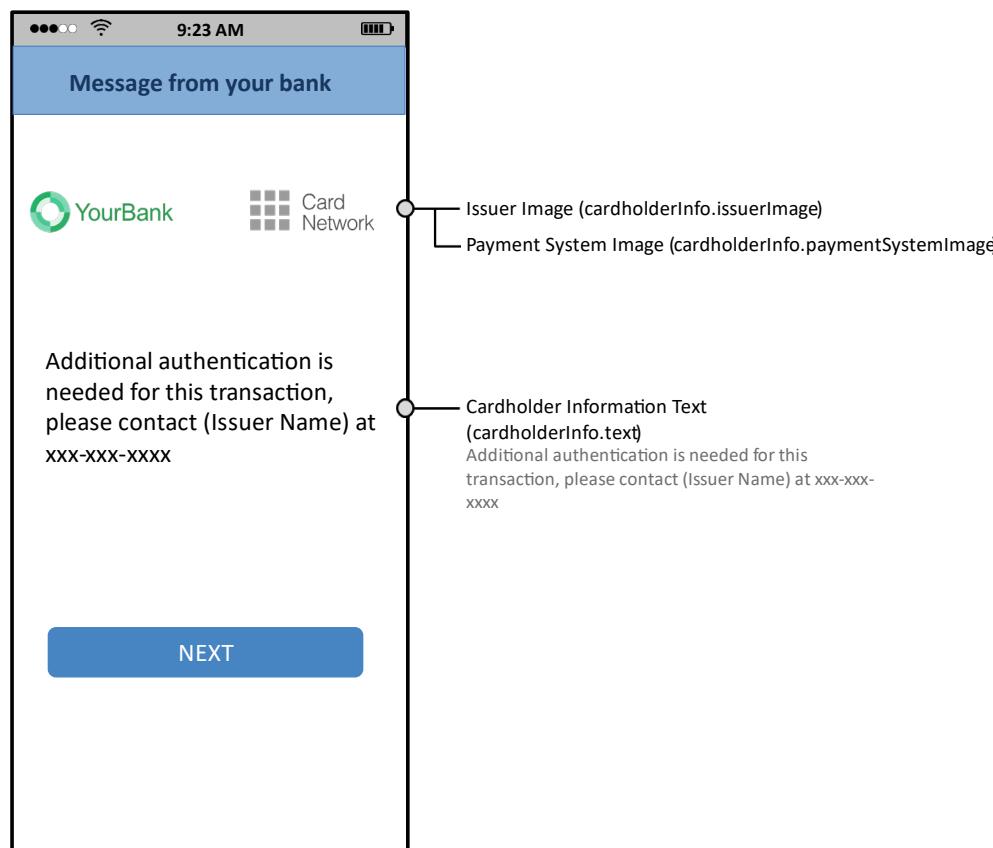
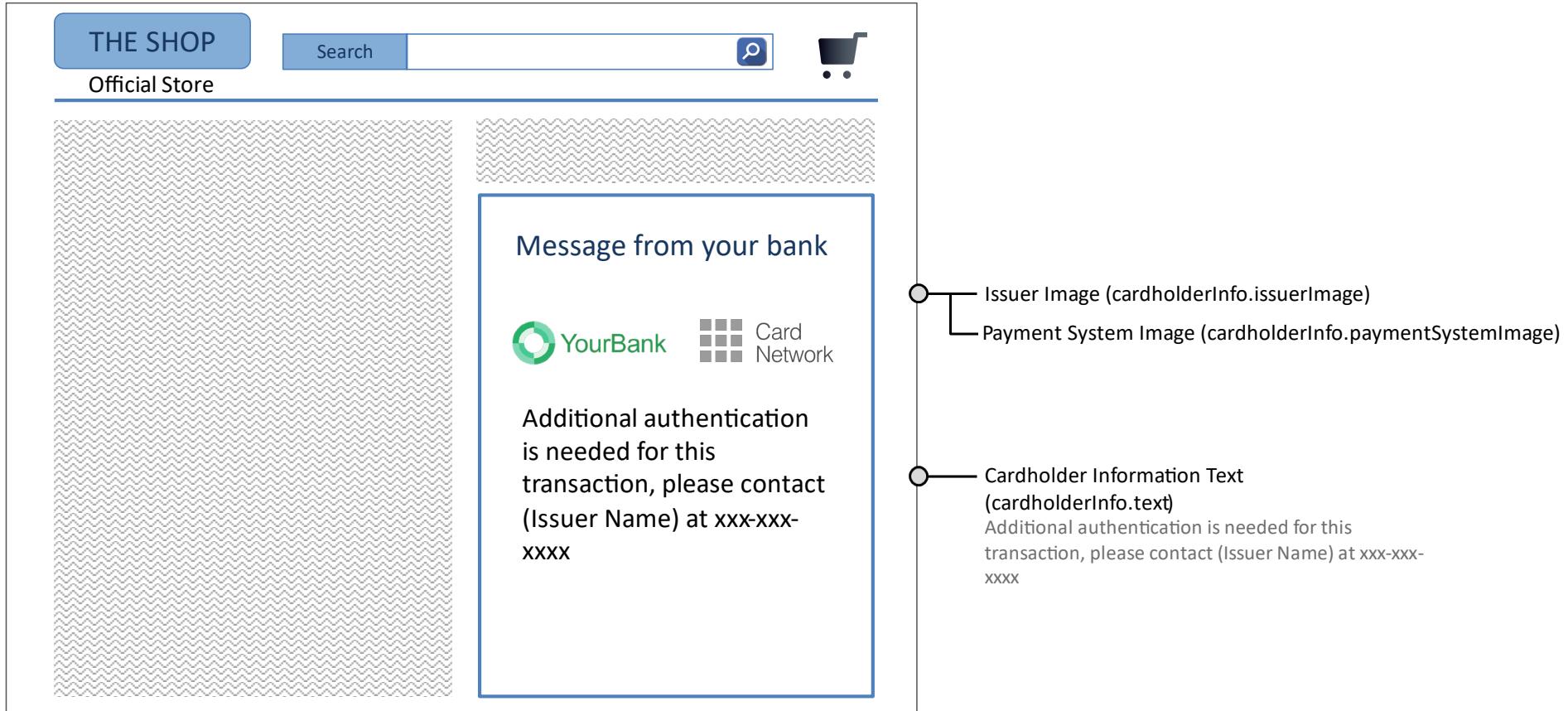


Figure A.2 Sample Cardholder Information UI Example—Browser—Landscape



A.21 SPC Transaction Data

The field names of the SPC Transaction Data match the names used in the SPC API (refer to [SPC API](#) for additional information).

Table A.28: SPC Transaction Data

Data Element/Field Name	Description	Source	Length/Format/Values	Inclusion
Additional Data Field Name: additionalData	For SPC API enhancement, to be defined in a future 3DS specification release	ACS	Length: Variable, maximum 90000 characters JSON Data Type: Object	ARes = O
Challenge Field Name: challenge	Random string generated by the ACS to prevent replay attacks.	ACS	Length: Variable, 43–100 characters JSON Data Type: String Base64url-encoded Example: a random 32-byte value that has been Base64url-encoded gives a 43-character string.	ARes = R
Challenge Information Text Field Name: challengeInfoText	Text provided by the ACS to be displayed during the SPC authentication.	ACS	Length: Variable, maximum 350 characters JSON Data Type: String	ARes = C Required when supported by the SPC API
Currency Field Name: currency	Transaction amount currency to be displayed during the SPC authentication	ACS	Length: 3 characters JSON Data Type: String Values accepted: <ul style="list-style-type: none">• ISO 4217 three-digit currency codes, other than those listed in Table A.5.	ARes = R

Data Element/Field Name	Description	Source	Length/Format/Values	Inclusion
Display Name Field Name: displayName	Card or product name (Payment Instrument) to be displayed during the SPC authentication.	ACS	Length: Variable, maximum 40 characters JSON Data Type: String	ARes = R
Icon Field Name: icon	Card image (Payment Instrument) URL or Data URL to be displayed during the SPC authentication.	ACS	Length: Variable, maximum 4096 characters JSON Data Type: String Values accepted: <ul style="list-style-type: none">• Fully Qualified URL or Data URL in correct JSON Object format<ul style="list-style-type: none">○ Fully Qualified URL: Variable length, maximum 2048 characters○ Data URL: Variable length, maximum 4096 characters. The Data URL embeds the image in Base64-encoded format.	ARes = R
Issuer Image SPC Field Name: issuerImageSpc	Issuer logo or Image URLs or Data URLs to be displayed during the SPC authentication. Includes at minimum the Default Image and at maximum the three Fully Qualified URLs or Data URLs defined as default, dark mode or monochrome images of the Issuer Image SPC. <ul style="list-style-type: none">• Default Image Field Name: default• Dark Mode Image Field Name: dark• Monochrome Image Field Name: monochrome	ACS	Length: Variable, maximum 90000 characters JSON Data Type: JSON object Values accepted: <ul style="list-style-type: none">• Fully Qualified URL or Data URL in correct JSON Object format<ul style="list-style-type: none">○ Fully Qualified URL: Variable length, maximum 2048 characters○ Data URL: Variable length, maximum 30000 characters. The Data URL embeds the image in Base64-encoded format. Refer to RFC 2397. If present, the Issuer Image SPC object should contain, at minimum, the Default Image.	ARes = C Required when supported by the SPC API

Data Element/Field Name	Description	Source	Length/Format/Values	Inclusion
	<p>Example Fully Qualified URL:</p> <pre>"issuerImageSpc":{ "default":"https://acs.com/ /defaultspcimage.png"}</pre> <p>Example Data URL:</p> <pre>"issuerImageSpc":{ "default":"data:image/png; base64,iVBORw0KGgoAA..."}</pre>			
Payee Name Field Name: payeeName	The display name of the payee that this SPC call is for (e.g. the Merchant). Matches the Merchant Name from the AReq message.	ACS	Length: Variable, maximum 40 characters JSON Data Type: String	ARes = C Required if Payee Origin is NOT present
Payee Origin Field Name: payeeOrigin	The origin of the payee that this SPC call is for (e.g. the Merchant). Matches the Payee Origin from the AReq message.	ACS	Length: Variable, maximum 2048 characters JSON Data Type: String Value accepted: <ul style="list-style-type: none">• Fully Qualified URL	ARes = C Required if Payee Name is NOT present
Payment System Image SPC Field Name: psImageSpc	Payment System logo or Image URLs to be displayed during the SPC authentication. Includes at minimum the Default Image and at maximum the three Fully Qualified URLs defined as default, dark mode or monochrome images of the Payment System Image SPC. <ul style="list-style-type: none">• Default Image Field Name: default	ACS	Length: Variable, maximum 90000 characters JSON Data Type: JSON object Values accepted: <ul style="list-style-type: none">• Fully Qualified URL or Data URL in correct JSON Object format<ul style="list-style-type: none">○ Fully Qualified URL: Variable length, maximum 2048 characters	ARes = C Required when supported by the SPC API

Data Element/Field Name	Description	Source	Length/Format/Values	Inclusion
	<ul style="list-style-type: none"> • Dark Mode Image Field Name: dark • Monochrome Image Field Name: monochrome <p>Example Fully Qualified URL:</p> <pre>"psImageSpc": { "default": "https://ds.com/defaultspcimage.png" }</pre> <p>Example Data URL:</p> <pre>"psImageSpc": { "default": "data:image/png; base64,c2RzYWRhc2Q..." }</pre>		<ul style="list-style-type: none"> ○ Data URL: Variable length, maximum 30000 characters. The Data URL embeds the image in Base64-encoded format. Refer to RFC 2397. <p>If present, the Payment System Image SPC object should contain, at minimum, the Default Image.</p>	
Timeout Field Name: timeout	The number of milliseconds before the request to sign the transaction details times out.	ACS	<p>Length: Variable, 5–6 characters JSON Data Type: String Value accepted:</p> <ul style="list-style-type: none"> • Integer coded as a string in the range 60000–500000 	ARes = R
Value Field Name: value	Transaction amount as a decimal value to be displayed during the SPC authentication.	ACS	<p>Length: Variable, maximum 40 characters JSON Data Type: String</p>	ARes = R

Data Element/Field Name	Description	Source	Length/Format/Values	Inclusion
WebAuthn SPC Extension Indicator Field Name: extInd	For SPC and WebAuthn API enhancement.	ACS	Length: 1 character JSON Data Type: String Value accepted: <ul style="list-style-type: none">• Y = Extension requested <p>Only present if value = Y</p>	ARes = O

A.22 HTTP Headers

Table A.29: HTTP Headers

Message	X-Request-ID	X-Response-ID
AReq (3DS Server → DS)	3DS Server Transaction ID	
AReq (DS → ACS)	DS Transaction ID	
ARes (ACS → DS)	DS Transaction ID	ACS Transaction ID
ARes (DS → 3DS Server)	3DS Server Transaction ID	DS Transaction ID
RReq (ACS → DS)	ACS Transaction ID	
RReq (DS → 3DS Server)	DS Transaction ID	
RRes (3DS Server → DS)	DS Transaction ID	3DS Server Transaction ID
RRes (DS → ACS)	ACS Transaction ID	DS Transaction ID
PReq (3DS Server → DS)	3DS Server Transaction ID	
PRes (DS → 3DS Server)	3DS Server Transaction ID	DS Transaction ID
OReq (DS → 3DS Server or ACS)	DS Transaction ID	
ORes (3DS Server or ACS → DS)	DS Transaction ID	3DS Server Transaction ID or ACS Transaction ID
CReq (SDK → ACS)	SDK Transaction ID	
CRes (ACS → SDK)	SDK Transaction ID	ACS Transaction ID

HTTP Header Examples

- Request message:

X-Request-ID: d07c7b7f-a987-4401-a76a-b1ad60d07837

- Response message:

X-Request-ID: d07c7b7f-a987-4401-a76a-b1ad60d07837

X-Response-ID: 9fca3f1c-6178-4668-aed9-d5d3ff0b0d98

Annex B Message Format

This annex provides the EMV 3-D Secure data elements and field names by Message Type. Refer to Table A.1 for data element specifications.

B.1 AReq Message Data Elements

Table A.1 outlines the default validation requirements for the AReq message. A specific DS may specify other DS validations or actions to meet requirements specific for that DS.

Table B.1: AReq Data Elements

Data Element	Field Name
3DS Method Completion Indicator	threeDSCompInd
3DS Method ID	threeDSMethodId
3DS Requestor Authentication Indicator	threeDSRequestorAuthenticationInd
3DS Requestor Authentication Information	threeDSRequestorAuthenticationInfo
3DS Requestor Authentication Method Verification Indicator	threeDSReqAuthMethodInd
3DS Requestor Challenge Indicator	threeDSRequestorChallengeInd
3DS Requestor Decoupled Max Time	threeDSRequestorDecMaxTime
3DS Requestor Decoupled Request Indicator	threeDSRequestorDecReqInd
3DS Requestor ID	threeDSRequestorID
3DS Requestor Name	threeDSRequestorName
3DS Requestor Prior Transaction Authentication Information	threeDSRequestorPriorAuthenticationInfo
3DS Requestor SPC Support	threeDSRequestorSpcSupport
3DS Requestor URL	threeDSRequestorURL
3DS Server Operator ID	threeDSServerOperatorID
3DS Server Reference Number	threeDSServerRefNumber
3DS Server Transaction ID	threeDSServerTransID
3DS Server URL	threeDSServerURL

Data Element	Field Name
3RI Indicator	threeRIInd
Accept Language	acceptLanguage
Account Type	acctType
Acquirer BIN	acquirerBIN
Acquirer Country Code	acquirerCountryCode
Acquirer Country Code Source	acquirerCountryCodeSource
Acquirer Merchant ID	acquirerMerchantID
Address Match Indicator	addrMatch
App IP Address	appIp
Broadcast Information	broadInfo
Browser Accept Headers	browserAcceptHeader
Browser IP Address	browserIP
Browser Java Enabled	browserJavaEnabled
Browser JavaScript Enabled	browserJavascriptEnabled
Browser Language	browserLanguage
Browser Screen Color Depth	browserColorDepth
Browser Screen Height	browserScreenHeight
Browser Screen Width	browserScreenWidth
Browser Time Zone	browserTZ
Browser User-Agent	browserUserAgent
Browser User Device ID	deviceId
Browser User ID	userId
Card Security Code	cardSecurityCode
Card Security Code Status Source	cardSecurityCodeStatusSource
Card Security Code Status	cardSecurityCodeStatus
Card/Token Expiry Date	cardExpiryDate
Cardholder Account Identifier	acctID

Data Element	Field Name
Cardholder Account Information	acctInfo
Cardholder Account Number	acctNumber
Cardholder Billing Address City	billAddrCity
Cardholder Billing Address Country	billAddrCountry
Cardholder Billing Address Line 1	billAddrLine1
Cardholder Billing Address Line 2	billAddrLine2
Cardholder Billing Address Line 3	billAddrLine3
Cardholder Billing Address Postal Code	billAddrPostCode
Cardholder Billing Address State	billAddrState
Cardholder Email Address	email
Cardholder Home Phone Number	homePhone
Cardholder Mobile Phone Number	mobilePhone
Cardholder Name	cardholderName
Cardholder Shipping Address City	shipAddrCity
Cardholder Shipping Address Country	shipAddrCountry
Cardholder Shipping Address Line 1	shipAddrLine1
Cardholder Shipping Address Line 2	shipAddrLine2
Cardholder Shipping Address Line 3	shipAddrLine3
Cardholder Shipping Address Postal Code	shipAddrPostCode
Cardholder Shipping Address State	shipAddrState
Cardholder Work Phone Number	workPhone
Default-SDK Type	defaultSdkType
Device Binding Status	deviceBindingStatus
Device Binding Status Source	deviceBindingStatusSource
Device Channel	deviceChannel
Device Information	deviceInfo
Device Rendering Options Supported	deviceRenderOptions

Data Element	Field Name
DS Reference Number	dsReferenceNumber
DS Transaction ID	dsTransID
DS URL	dsURL
EMV Payment Token Indicator	payTokenInd
EMV Payment Token Information	payTokenInfo
EMV Payment Token Source	payTokenSource
Instalment Payment Data	purchaseInstalData
Merchant Category Code	mcc
Merchant Country Code	merchantCountryCode
Merchant Name	merchantName
Merchant Risk Indicator	merchantRiskIndicator
Message Category	messageCategory
Message Extension	messageExtension
Message Type	messageType
Message Version Number	messageVersion
Multi-Transaction	multiTransaction
Notification URL	notificationURL
Payee Origin	payeeOrigin
Purchase Amount	purchaseAmount
Purchase Currency	purchaseCurrency
Purchase Currency Exponent	purchaseExponent
Purchase Date & Time	purchaseDate
Recurring Amount	recurringAmount
Recurring Currency	recurringCurrency
Recurring Currency Exponent	recurringExponent
Recurring Date	recurringDate
Recurring Expiry	recurringExpiry

Data Element	Field Name
Recurring Frequency	recurringFrequency
Recurring Indicator	recurringInd
SDK App ID	sdkAppID
SDK Encrypted Data	sdkEncData
SDK Ephemeral Public Key (Qc)	sdkEphemPubKey
SDK Maximum Timeout	sdkMaxTimeout
SDK Reference Number	sdkReferenceNumber
SDK Server Signed Content	sdkServerSignedContent
SDK Transaction ID	sdkTransID
SDK Type	sdkType
Seller Information	sellerInfo
SPC Incompletion Indicator	spcIncompInd
Split-SDK Type	splitSdkType
Tax ID	taxId
Transaction Type	transType
Trust List Status	trustListStatus
Trust List Status Source	trustListStatusSource

B.2 ARes Message Data Elements

Table A.1 outlines the default validation requirements for the ARes message. A specific DS may specify other DS validations or actions to meet requirements specific for that DS.

Table B.2: ARes Data Elements

Data Element	Field Name
3DS Requestor App URL Indicator	threeDSRequestorAppURLInd
3DS Server Transaction ID	threeDSServerTransID
ACS Challenge Mandated Indicator	acsChallengeMandated
ACS Decoupled Confirmation Indicator	acsDecConInd
ACS Operator ID	acsOperatorID
ACS Reference Number	acsReferenceNumber
ACS Rendering Type	acsRenderingType
ACS Signed Content	acsSignedContent
ACS Transaction ID	acsTransID
ACS URL	acsURL
Authentication Method	authenticationMethod
Authentication Value	authenticationValue
Broadcast Information	broadInfo
Cardholder Information Text	cardholderInfo
Card Security Code Status Source	cardSecurityCodeStatusSource
Card Security Code Status	cardSecurityCodeStatus
Device Binding Status	deviceBindingStatus
Device Binding Status Source	deviceBindingStatusSource
Device Information Recognised Version	deviceInfoRecognisedVersion
DS Reference Number	dsReferenceNumber
DS Transaction ID	dsTransID
Electronic Commerce Indicator	eci

Data Element	Field Name
Message Extension	messageExtension
Message Type	messageType
Message Version Number	messageVersion
SDK Transaction ID	sdkTransID
SPC Transaction Data	spcTransData
Transaction Challenge Exemption	transChallengeExemption
Transaction Status	transStatus
Transaction Status Reason	transStatusReason
Transaction Status Reason Information	transStatusReasonInfo
Trust List Status	trustListStatus
Trust List Status Source	trustListStatusSource
WebAuthn Credential List	webAuthnCredList

B.3 CReq Message Data Elements

Table A.1 outlines the default validation requirements for the CReq message.

Table B.3: CReq Data Elements

Data Element	Field Name
3DS Requestor App URL	threeDSRequestorAppURL
3DS Server Transaction ID	threeDSServerTransID
ACS Transaction ID	acsTransID
Challenge Additional Code	challengeAddCode
Challenge Cancelation Indicator	challengeCancel
Challenge Data Entry	challengeDataEntry
Challenge Data Entry 2	challengeDataEntryTwo
Challenge HTML Data Entry	challengeHTMLDataEntry
Challenge No Entry	challengeNoEntry
Challenge Window Size	challengeWindowSize
Device Binding Data Entry	deviceBindingDataEntry
Information Continuation Indicator	infoContinueIndicator
Message Extension	messageExtension
Message Type	messageType
Message Version Number	messageVersion
OOB App Status	oobAppStatus
OOB App URL Indicator	oobAppURLInd
OOB Continuation Indicator	oobContinue
Resend Challenge Information Code	resendChallenge
SDK Counter SDK to ACS	sdkCounterStoA
SDK Transaction ID	sdkTransID
Trust List Data Entry	trustListDataEntry

B.4 CRes Message Data Elements

Table A.1 outlines the default validation requirements for the CRes message.

Table B.4: CRes Data Elements

Data Element	Field Name
3DS Server Transaction ID	threeDSServerTransID
ACS Counter ACS to SDK	acsCounterAtoS
ACS Transaction ID	acsTransID
ACS HTML	acsHTML
ACS UI Type	acsUiType
Challenge Additional Label	challengeAddLabel
Challenge Completion Indicator	challengeCompletionInd
Challenge Entry Box	challengeEntryBox
Challenge Entry Box 2	challengeEntryBoxTwo
Challenge Information Header	challengeInfoHeader
Challenge Information Label	challengeInfoLabel
Challenge Information Text	challengeInfoText
Challenge Information Text Indicator	challengeInfoTextIndicator
Challenge Selection Information	challengeSelectInfo
Device Binding Information Text	deviceBindingInfoText
Expandable Information Label	expandInfoLabel
Expandable Information Text	expandInfoText
Information Continuation Label	infoContinueLabel
Issuer Image	issuerImage
Message Extension	messageExtension
Message Type	messageType
Message Version Number	messageVersion
OOB App Label	oobAppLabel

Data Element	Field Name
OOB App URL	oobAppURL
OOB Continuation Label	oobContinueLabel
Payment System Image	psImage
Resend Information Label	resendInformationLabel
SDK Transaction ID	sdkTransID
Submit Authentication Label	submitAuthenticationLabel
Toggle Position Indicator	togglePositionInd
Trust List Information Text	trustListInfoText
Why Information Label	whyInfoLabel
Why Information Text	whyInfoText

B.5 Final CRes Message Data Elements

Table B.5 provides the data elements for the final CRes message sent upon completion of the challenge from the ACS.

Table A.1 outlines the default validation requirements for the CRes message.

Table B.5: Final CRes Data Elements

Data Element	Field Description
3DS Server Transaction ID	threeDSServerTransID
ACS Counter ACS to SDK	acsCounterAtoS
ACS Transaction ID	acsTransID
Challenge Completion Indicator	challengeCompletionInd
Message Extension	messageExtension
Message Type	messageType
Message Version Number	messageVersion
SDK Transaction ID	sdkTransID
Transaction Status	transStatus

B.6 PReq Message Data Elements

Table A.1 outlines the default validation requirements for the PReq message.

Table B.6: PReq Data Elements

Data Element	Field Name
3DS Server Operator ID	threeDSServerOperatorID
3DS Server Reference Number	threeDSServerRefNumber
3DS Server Transaction ID	threeDSServerTransID
Card Range Data Download Indicator	cardRangeDataDownloadInd
Message Extension	messageExtension
Message Type	messageType
Message Version Number	messageVersion
Serial Number	serialNum

B.7 PRes Message Data Elements

Table A.1 outlines the default validation requirements for the PRes message.

Table B.7: PRes Data Elements

Data Element	Field Name
3DS Server Transaction ID	threeDSServerTransID
Card Range Data	cardRangeData
Card Range Data File URL	cardRangeDataFileURL
DS Protocol Versions	dsProtocolVersions
DS Transaction ID	dsTransID
DS URL List	dsUrlList
Message Extension	messageExtension
Message Type	messageType
Message Version Number	messageVersion
Read Order	readOrder
Serial Number	serialNum

B.8 RReq Message Data Elements

Table A.1 outlines the default validation requirements for the RReq message.

Table B.8: RReq Data Elements

Data Element	Field Name
3DS Server Transaction ID	threeDSServerTransID
ACS Rendering Type	acsRenderingType
ACS Transaction ID	acsTransID
Authentication Method	authenticationMethod
Authentication Value	authenticationValue
Cardholder Information Text	cardholderInfo
Challenge Cancelation Indicator	challengeCancel
Challenge Error Reporting	challengeErrorReporting
Device Binding Status	deviceBindingStatus
Device Binding Status Source	deviceBindingStatusSource
DS Transaction ID	dsTransID
Electronic Commerce Indicator	eci
Interaction Counter	interactionCounter
Message Category	messageCategory
Message Extension	messageExtension
Message Type	messageType
Message Version Number	messageVersion
SDK Transaction ID	sdkTransID
Transaction Status	transStatus
Transaction Status Reason	transStatusReason
Transaction Status Reason Information	transStatusReasonInfo
Trust List Status	trustListStatus
Trust List Status Source	trustListStatusSource

B.9 RRes Message Data Elements

Table A.1 outlines the default validation requirements for the RRes message.

Table B.9: RRes Data Elements

Data Element	Field Name
3DS Server Transaction ID	threeDSserverTransID
ACS Transaction ID	acsTransID
DS Transaction ID	dsTransID
Message Extension	messageExtension
Message Type	messageType
Message Version Number	messageVersion
Results Message Status	resultsStatus
SDK Transaction ID	sdkTransID

B.10 OReq Message Data Elements

Table A.1 outlines the default validation requirements for the OReq message.

Table B.10: OReq Data Elements

Data Element	Field Name
DS Reference Number	dsReferenceNumber
DS Transaction ID	dsTransID
Message Extension	messageExtension
Message Type	messageType
Message Version Number	messageVersion
Operation Category	opCategory
Operation Description	opDescription
Operation Expiration Date	opExpDate
Operation Prior Transaction Reference	opPriorTransRef
Operation Sequence	opSeq
Operation Severity	opSeverity

B.11 ORes Message Data Elements

Table A.1 outlines the default validation requirements for the ORes message.

Table B.11: ORes Data Elements

Data Element	Field Name
3DS Server Reference Number	threeDSServerRefNumber
3DS Server Transaction ID	threeDSServerTransID
ACS Reference Number	acsReferenceNumber
ACS Transaction ID	acsTransID
DS Transaction ID	dsTransID
Message Extension	messageExtension
Message Type	messageType
Message Version Number	messageVersion
Operation Message Status	opStatus

B.12 Error Messages Data Elements

Table A.4 provides detailed information including Error Code values.

Table B.12: Error Message Data Elements

Data Element	Field Name
3DS Server Transaction ID	threeDSServerTransID
ACS Transaction ID	acsTransID
DS Transaction ID	dsTransID
Error Code	errorCode
Error Component	errorComponent
Error Description	errorDescription
Error Detail	errorDetail
Error Message Type	errorMessageType
Message Type	messageType
Message Version Number	messageVersion
SDK Transaction ID	sdkTransID

Annex C Generate ECC Key Pair

A number of available cryptographic libraries include ECC key pair generation as a function. If not available, it is recommended to use a method of ECC key pair generation following [ISO/IEC 15946-1].

In this method, a random number is obtained and tested to determine that it will produce a value of d (the private key) in the correct range ($1 < d < n$). If d is out-of-range, another random number is obtained (for example, the process is iterated until an acceptable value of d is obtained).

Note: The integers used in this function are large (32 or 66 bytes), which may mean they would be represented as strings of bytes (as in most cryptographic libraries) rather than built-in integers in an implementation.

The following process or its equivalent may be used to generate an ECC key pair.

C.1 Input

Curve parameters (p, a, b, G, n, h):

C.2 Output

- status—Status returned from the key pair generation procedure. The status will indicate SUCCESS or an ERROR.
 - (d, Q) —Generated private and public key
 - d , the generated private key, is an integer in the range $[1, n-1]$
 - Q , the generated public key, is the point on the specified curve
- If an error is encountered during the generation process, no values for d and Q should be returned

C.2.1 Process

1. $N := \text{len}(n)$, the bit-length of n
2. $d := \text{StringToInteger}(\text{Random}(N))$
3. If $(d > n - 2)$, then go to step 2
4. $d := d + 1$
5. $Q := \text{PointMultiply}(d, G, \text{CurveID})$
6. If successful, return SUCCESS, d , and Q
Else, return ERROR status.

Auxiliary functions used:

- `StringToInteger(s)` converts a string `s` of bits to a non-negative integer
- `Random(c)` generates a string of `c` bits, where `c` is a positive integer
- `PointMultiply()` performs scalar multiplication

Annex D Approved Transport Layer Security Versions

For establishing the links secured by TLS between the DS, 3DS Server, ACS and 3DS SDK, the version number shall be V1.2 or higher.

RSA keys shall be 2048 bits or longer.

ECC keys shall be 256 bits or longer.

Requirements and recommendations for the supported Cipher Suites are as follows:

D.1 Cipher Suites for TLS 1.2

D.1.1 Supported Cipher Suites

- TLS_ECDHE_RSA_WITH_AES_128_GCM_SHA256
- TLS_ECDHE_ECDSA_WITH_AES_128_GCM_SHA256

Curve P-256 shall be used and indicated in the cipher suite extension.

3DS Server, ACS and 3DS SDK shall be able to support both cipher suites.

DS shall be able to support at least one of the cipher suites.

DS CA shall provide client and server certificates for the cipher suite(s) supported by the DS.

D.1.2 Other Cipher Suites

If required for any other reason or for legacy purposes, additional cipher suites may be supported. Interoperability testing is at the implementer's discretion.

D.2 Not Supported

The following cipher suites shall not be presented or accepted:

- Any cipher suite represented as 'Null', 'Anonymous/Anon'
- Any cipher suite incorporating any of the algorithms 'RC2', 'RC4', 'DES', 'IDEA', 'KRB5', 'ARIA', or 'MD5'
- Any cipher suite incorporating an export grade algorithm using 'EXPORT'.

Note: 3DES and SHA-1 are to be phased out and may become unsupported algorithms in future versions of this specification.

*This page intentionally left blank.

Requirements

[Req 1].....	52
[Req 2].....	52
[Req 3].....	52
[Req 4].....	52
[Req 5].....	52
[Req 6].....	52
[Req 7].....	52
[Req 8].....	53
[Req 9].....	53
[Req 10].....	53
[Req 11].....	51
[Req 12].....	53
[Req 13].....	53
[Req 14].....	53
[Req 15].....	53
[Req 16].....	53
[Req 17].....	54
[Req 18].....	54
[Req 19].....	54
[Req 20].....	54
[Req 21].....	54
[Req 22].....	54
[Req 23].....	54
[Req 24].....	55
[Req 25].....	55
[Req 26].....	55
[Req 27].....	55
[Req 28].....	55
[Req 29].....	55
[Req 30].....	56
[Req 31].....	56
[Req 32].....	56
[Req 33].....	57
[Req 34].....	57
[Req 35].....	57

[Req 36].....	57
[Req 37].....	58
[Req 38].....	58
[Req 39].....	58
[Req 40].....	58
[Req 41].....	58
[Req 42].....	59
[Req 43].....	59
[Req 44].....	59
[Req 45].....	59
[Req 46].....	59
[Req 47].....	59
[Req 48].....	59
[Req 49].....	59
[Req 50].....	59
[Req 51].....	60
[Req 52].....	60
[Req 53].....	60
[Req 54].....	60
[Req 55].....	60
[Req 56].....	60
[Req 57].....	60
[Req 58].....	60
[Req 59].....	60
[Req 60].....	61
[Req 61].....	61
[Req 62].....	62
[Req 63].....	62
[Req 64].....	62
[Req 65].....	62
[Req 66].....	62
[Req 67].....	62
[Req 68].....	62
[Req 69].....	62
[Req 70].....	63
[Req 71].....	63
[Req 72].....	63

[Req 73].....	63
[Req 74].....	63
[Req 75].....	63
[Req 76].....	63
[Req 77].....	63
[Req 78].....	64
[Req 79].....	64
[Req 80].....	70
[Req 81].....	70
[Req 82].....	70
[Req 83].....	70
[Req 84].....	70
[Req 85].....	70
[Req 86].....	71
[Req 87].....	71
[Req 88].....	71
[Req 89].....	71
[Req 90].....	71
[Req 91].....	72
[Req 92].....	72
[Req 93].....	72
[Req 94].....	72
[Req 95].....	72
[Req 96].....	72
[Req 97].....	72
[Req 98].....	73
[Req 99].....	73
[Req 100].....	73
[Req 101].....	73
[Req 102].....	73
[Req 103].....	73
[Req 104].....	73
[Req 105].....	74
[Req 106].....	74
[Req 107].....	74
[Req 108].....	74
[Req 109].....	75

[Req 110]	75
[Req 111]	75
[Req 112]	76
[Req 113]	76
[Req 114]	76
[Req 115]	76
[Req 116]	76
[Req 117]	76
[Req 118]	77
[Req 119]	77
[Req 120]	77
[Req 121]	77
[Req 122]	78
[Req 123]	78
[Req 124]	79
[Req 125]	79
[Req 126]	79
[Req 127]	79
[Req 128]	79
[Req 129]	80
[Req 130]	80
[Req 131]	80
[Req 132]	80
[Req 133]	80
[Req 134]	80
[Req 135]	80
[Req 136]	80
[Req 137]	81
[Req 138]	81
[Req 139]	81
[Req 140]	81
[Req 141]	100
[Req 142]	100
[Req 143]	100
[Req 144]	101
[Req 145]	101
[Req 146]	101

[Req 147]	101
[Req 148]	101
[Req 151]	101
[Req 153]	123
[Req 154]	123
[Req 155]	124
[Req 156]	124
[Req 157]	124
[Req 158]	124
[Req 159]	135
[Req 160]	135
[Req 161]	135
[Req 162]	135
[Req 163]	135
[Req 164]	135
[Req 165]	135
[Req 166]	136
[Req 167]	136
[Req 168]	136
[Req 169]	136
[Req 170]	136
[Req 171]	136
[Req 172]	137
[Req 173]	137
[Req 174]	137
[Req 175]	137
[Req 176]	137
[Req 177]	137
[Req 178]	137
[Req 179]	137
[Req 180]	137
[Req 181]	137
[Req 182]	137
[Req 183]	137
[Req 184]	145
[Req 185]	145
[Req 186]	145

[Req 187]	145
[Req 188]	145
[Req 189]	145
[Req 190]	145
[Req 191]	146
[Req 192]	146
[Req 193]	146
[Req 195]	146
[Req 196]	147
[Req 197]	147
[Req 198]	147
[Req 199]	147
[Req 200]	147
[Req 201]	147
[Req 202]	147
[Req 203]	147
[Req 204]	148
[Req 205]	148
[Req 206]	148
[Req 207]	148
[Req 208]	148
[Req 209]	148
[Req 210]	149
[Req 212]	149
[Req 213]	149
[Req 215]	149
[Req 216]	150
[Req 217]	150
[Req 218]	150
[Req 219]	150
[Req 220]	150
[Req 221]	151
[Req 222]	151
[Req 223]	151
[Req 224]	151
[Req 225]	151
[Req 226]	151

[Req 227]	151
[Req 228]	152
[Req 229]	152
[Req 231]	152
[Req 232]	152
[Req 233]	153
[Req 234]	153
[Req 235]	153
[Req 236]	153
[Req 237]	153
[Req 239]	153
[Req 240]	154
[Req 241]	154
[Req 242]	154
[Req 243]	154
[Req 244]	154
[Req 245]	154
[Req 246]	155
[Req 247]	155
[Req 248]	155
[Req 249]	155
[Req 250]	156
[Req 253]	157
[Req 254]	157
[Req 255]	158
[Req 256]	158
[Req 257]	158
[Req 258]	158
[Req 259]	158
[Req 260]	159
[Req 261]	159
[Req 262]	159
[Req 263]	159
[Req 264]	159
[Req 265]	160
[Req 266]	160
[Req 267]	160

[Req 268].....	160
[Req 269].....	160
[Req 270].....	160
[Req 271].....	83
[Req 272].....	83
[Req 273].....	83
[Req 274].....	83
[Req 275].....	83
[Req 276].....	84
[Req 277].....	84
[Req 278].....	84
[Req 279].....	84
[Req 280].....	84
[Req 281].....	84
[Req 282].....	84
[Req 283].....	85
[Req 285].....	85
[Req 286].....	85
[Req 287].....	85
[Req 288].....	85
[Req 289].....	85
[Req 290].....	85
[Req 291].....	85
[Req 292].....	86
[Req 293].....	86
[Req 294].....	86
[Req 295].....	87
[Req 296].....	87
[Req 297].....	87
[Req 298].....	87
[Req 299].....	87
[Req 300].....	53
[Req 301].....	71
[Req 302].....	83
[Req 303].....	155
[Req 304].....	156
[Req 305].....	57

[Req 306]	75
[Req 307]	78
[Req 308]	87
[Req 309]	148
[Req 310]	61
[Req 311]	147
[Req 312]	153
[Req 313]	145
[Req 314]	96
[Req 315]	159
[Req 318]	55
[Req 319]	73
[Req 320]	147
[Req 321]	57
[Req 322]	57
[Req 323]	58
[Req 324]	160
[Req 325]	75
[Req 326]	75
[Req 327]	77
[Req 328]	86
[Req 330]	87
[Req 332]	88
[Req 333]	88
[Req 334]	88
[Req 335]	88
[Req 336]	88
[Req 337]	88
[Req 338]	89
[Req 339]	89
[Req 340]	89
[Req 341]	89
[Req 342]	96
[Req 344]	151
[Req 345]	62
[Req 346]	63
[Req 347]	79

[Req 348].....	80
[Req 349].....	88
[Req 350].....	88
[Req 351].....	88
[Req 352].....	88
[Req 353].....	88
[Req 354].....	88
[Req 355].....	58
[Req 356].....	77
[Req 357].....	87
[Req 358].....	96
[Req 359].....	96
[Req 362].....	105
[Req 363].....	105
[Req 364].....	105
[Req 365].....	105
[Req 366].....	105
[Req 367].....	105
[Req 368].....	105
[Req 369].....	105
[Req 370].....	106
[Req 371].....	125
[Req 372].....	125
[Req 373].....	125
[Req 374].....	125
[Req 375].....	125
[Req 376].....	125
[Req 377].....	125
[Req 378].....	125
[Req 380].....	138
[Req 381].....	138
[Req 382].....	138
[Req 383].....	138
[Req 384].....	138
[Req 385].....	156
[Req 386].....	55
[Req 387].....	106

[Req 388].....	101
[Req 389].....	101
[Req 390].....	53
[Req 391].....	96
[Req 392].....	105
[Req 393].....	136
[Req 394].....	53
[Req 395].....	96
[Req 396].....	147
[Req 397].....	147
[Req 398].....	105
[Req 399].....	66
[Req 400].....	66
[Req 401].....	66
[Req 402].....	67
[Req 403].....	67
[Req 404].....	67
[Req 405].....	67
[Req 406].....	67
[Req 407].....	67
[Req 408].....	68
[Req 409].....	68
[Req 410].....	73
[Req 411].....	76
[Req 412].....	87
[Req 413].....	136
[Req 415].....	159
[Req 416].....	169
[Req 417].....	169
[Req 418].....	96
[Req 419].....	51
[Req 420].....	53
[Req 421].....	58
[Req 422].....	72
[Req 423].....	83
[Req 424].....	153
[Req 425].....	155

[Req 426]	156
[Req 427]	85
[Req 428]	155
[Req 429]	106
[Req 430]	147
[Req 431]	148
[Req 432]	148
[Req 433]	148
[Req 434]	149
[Req 435]	170
[Req 436]	170
[Req 437]	170
[Req 438]	170
[Req 439]	170
[Req 440]	170
[Req 441]	71
[Req 442]	77
[Req 443]	91
[Req 444]	91
[Req 445]	106
[Req 446]	105
[Req 447]	90
[Req 448]	90
[Req 449]	90
[Req 450]	91
[Req 451]	92
[Req 452]	151
[Req 453]	152
[Req 454]	152
[Req 455]	152
[Req 456]	155
[Req 457]	156
[Req 458]	156
[Req 459]	156
[Req 460]	156
[Req 461]	61
[Req 462]	62

[Req 463]	63
[Req 464]	78
[Req 465]	79
[Req 466]	80
[Req 467]	84
[Req 468]	146
[Req 469]	146