Technical integration guide (PSP)

> Important: The following functionalities are not yet available on the PAY.ON test or live gateway!

Setup of new 3D Secure 2.0 protocol

Using the 3D Secure 2.0 worklfow powered by the ACI UP ecommerce gateway, following changes must be done in the Business Intelligence Platform (BIP).

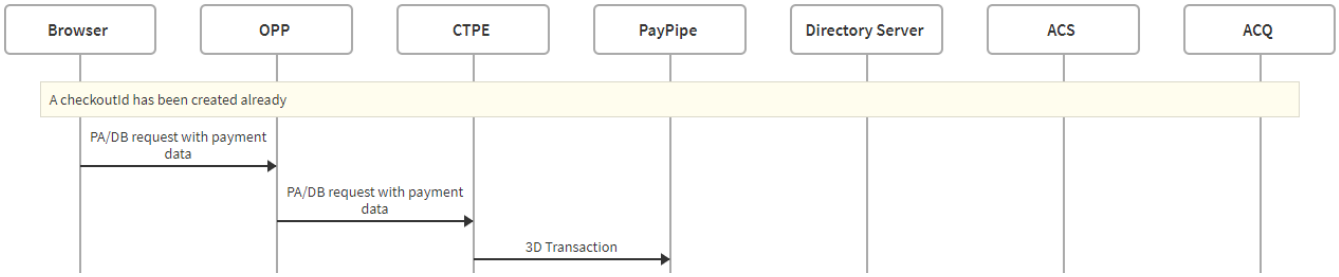
1. Enable 3D Secure 2.0 by changing the MPI type.
2. Configure a fallback strategy
   Go to "Risk-Management -> Risk Checks -> External checks -> 3D Secure settings"
   Change the setting of "Fallback to 3D Secure 1.0"
    1. true
    2. false (default)
3. 3D Secure 2.0 also requires two additional merchant data to be sent.
   Go to "Administration -> Account Data -> Available merchant accounts" and add the new required field to the merchant account of your choice

Once you set up , the next step depends on the type of integration you have with ACI.

1. OPP COPYandPAY
2. OPP Server to Server
3. XML Server to Server
4. OPP Server to Server - Standalone 3D Secure

3D Secure 2.0 workflows

Please find the full 3D Secure 2.0 workflow here.



Depending on the ´platform setting "Enable 3D Secure 2.0",

- Yes, 3D Secure 2.0 only
- Yes, 3D Secure 2.0 with to 1.0 if not supported

and taking it for granted that the card number of the holder is enrolled for 3D Secure,

ACI will check whether the 3D Secure is supported by the issuer and depending on the Platform setting "Enable 3D Secure 2.0, following workflow will get executed:

	Issuer support 3D Secure 2.0		**Issuer doesn't support 3D Secure 2.0**	
	No user redirect	User redirect (challenge)	No user redirect	User redirect (challenge)
Yes, 3D Secure 2.0 only	• RM.3D will be in a final state immediately. No further cardholder interaction is required. • The already provided data are sufficient for doing a proper 3D Secure risk assessment. • Continue with payment authorization in case of RM.3D is successful	• RM.3D will be in a pending state and further risk assessment is required. • The cardholder will get redirect to the issuers challengeUrl for two factor authentication. • RM.3D will get updated • Continue with payment authorization in case of RM.3D is successful	• RM.3D will be denied immediately • No payment authorization	
Yes, 3D Secure 2.0 with fallback to 1.0 if not supported	-			• RM.3D will be denied immediately • ACI automatically creates a second RM.3D for connecting to 3D Secure 1.0 • Continue with payment authorization in case of second RM.3D is successful ACI customer will be charged for two RM.3D transactions.

OPP COPYandPAY

For users of COPYandPAY, minimal additional effort is required compared to the current integration. The workflow is identical to the current 3D Secure 1.0 implementation. OPP COPYandPAY will handle the entire additional communication and will be responsible for collecting required browser based information automatically. Following data must be collected by the merchant and send in via

- Server/Server, create checkoutId
- being collected by adding additional input fields to the payment widget

Field name	**Mandatory/Optional**	**Source when using the CopyAndPay widget**
billing.city	Required unless market or regional mandate restricts sending this information.	Cardholder (via widget or API)
billing.country	Required unless market or regional mandate restricts sending this information.	Cardholder (via widget or API)
billing.street1	Required unless market or regional mandate restricts sending this information.	Cardholder (via widget or API)
billing.postcode	Required unless market or regional mandate restricts sending this information.	Cardholder (via widget or API)
customer.email	Required unless market or regional mandate restricts sending this information.	Cardholder (via widget or API)

customer.givenName	Required unless market or regional mandate restricts sending this information.	Cardholder (via widget or API)
customer.surname	Required unless market or regional mandate restricts sending this information.	Cardholder (via widget or API)

OPP Server to Server

For users who are integrating with the ACI UP ecommerce gateway via server to server will need to follow EMVCo's guidelines on the frontend integration. Please follow the steps described below:

1. Prepare your front-end and follow EMVCo's recommendation (see Section 4 "EMV 3-D Secure User Interface Templates, Requirements, and Guidelines" here) on how the authentication window should be shown in the webshop (eg. in and iframe or in a lightbox).
2. Prepare your back-end and send following additional information to OPP along with the payment information:

Field	Opp Field name	Description	Length/Format/Values
Accept header	customer.browser.acceptHeader	HTTP accept header sent from the cardholder's browser.	Length: Variable, maximum 2048 characters JSON Data Type: String Value accepted: If the total length of the accept header sent by the browser exceeds 2048 characters, the 3DS Server truncates
Language	customer.browser.language	The cardholder's browser language.	Length: Variable, 1–8 characters JSON Data Type: String
Screen height	customer.browser.screenHeight	This field contains the total height of the cardholder's screen in pixels.	Length: Variable, 1–6 characters JSON Data Type: String
Screen width	customer.browser.screenWidth	This field contains the total width of the cardholder's screen in pixels.	Length: Variable, 1–6 characters JSON Data Type: String
Browser timezone	customer.browser.timezone	This field contains the cardholder's browser local timezone.	Length: 1–5 characters JSON Data Type: String Value accepted: Value is returned from the getTimezoneOffset() method.
User agent	customer.browser.userAgent	This field contains the exact content of the HTTP User-Agent header.	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.
IP address	customer.browser.ipAddress	IP address of the cardholder's browser.	Length: Variable, maximum 45 characters JSON Data Type: String Value accepted: IPv4 address is represented in the dotted decimal format of 4 sets of decimal numbers separated by dots. The decimal number in each and every set is in the range 0 to 255. Example IPv4 address: 1.12.123.255 IPv6 address is represented as eight groups of four hexadecimal digits, each group representing 16 bits (two octets. The groups are separated by colons (:). Example IPv6 address: 2011:0db8:85 a3:0101:0101:8a2e: 03 70:7334
Java enabled	customer.browser.javaEnabled	true/false - Ability of the cardholder's browser to execute Java.	JSON Data Type: Boolean Values accepted: • true • false
Screen color depth	customer.browser.screenColorDepth	This field contains a value representing the bit depth of the color palette, in bits per pixel, for displaying images.	Length: 1–2 characters JSON Data Type: String Values accepted: 1 = 1 bit • 4 = 4 bits • 8 = 8 bits • 15 = 15 bits • 16 = 16 bits • 24 = 24 bits • 32 = 32 bits • 48 = 48 bits

<table>
<tr>
<td>Authentication window size</td>
<td>customer.browser.challengeWindow</td>
<td>Size of the authentication iframe which will render the ACS authentication front-end to the shopper for interaction.

Please send an Integer between 1-5. The integer corresponds to one the following resolutions:

<table>
<tr><th>1</th><th>2</th><th>3</th><th>4</th><th>5</th></tr>
<tr><td>250 x 400</td><td>390 x 400</td><td>500 x 60</td><td>600 x 400</td><td>Full screen</td></tr>
</table>
</td>
<td></td>
</tr>
</table>

Server to Server response

Server to Server response


```
{
        "id": "8ac7a4a0686138d701687eebfbc74747",
        "paymentType": "DB",
        "paymentBrand": "VISA",
        "result": {
                "code": "000.200.000",
                "description": "transaction pending"
        },
        "resultDetails": {
                "clearingInstituteName": "Elavon-euroconex_UK_Test"
        },
        "card": {
                "bin": "411111",
                "last4Digits": "1111",
                "holder": "Jane Jones",
                "expiryMonth": "05",
                "expiryYear": "2020"
        },
        "redirect": {
                "url": "https://test.oppwa.com/v1/threeDSecure/execute",
                "parameters": [{
                        "name": "name",
                        "value": "value"
                }],
                "preconditions": [{
                        "origin": "iframe#hidden",
                        "waitUntil": "iframe#onload",
                        "description": "Hidden iframe post for 3D Secure
2.0",
                        "method": "POST",
                        "url": "methodURL",
                        "parameters": [{
                                "name": "methodData",
                                "value": "methodData"
                        }]
                }]
        },
        "risk": {
                "score": "100"
        },
        "buildNumber": "deebd8c9af7d84ddee98c38b7f4afcc814012b5b@2019-01-
22 13:58:00 +0000",
        "timestamp": "2019-01-24 08:13:41+0000",
        "ndc":
"8a8294174b7ecb28014b9699220015ca_0557df43f75643d19479440642979e00"
}
```

XML Server to Server

For users who are integrating with the ACI UP ecommerce gateway via server to server will need to follow EMVCo's guidelines on the frontend integration. Please follow the steps described below:

1. Prepare your front-end and follow EMVCo's recommendation (see Section 4 "EMV 3-D Secure User Interface Templates, Requirements, and Guidelines" here) on how the authentication window should be shown in the webshop (eg. in and iframe or in a lightbox).
2. Prepare your back-end and send following additional information to OPP along with the payment information:

<table>
<tr><th>Field</th><th>XML Field name</th><th>Description</th><th>Length/Format/Values</th></tr>
<tr><td>Accept header</td><td rowspan="9"><pre><Browser>
   <AcceptHeader>text/html<
/AcceptHeader>
   <Language>de</Language>
   <ScreenHeight>480<
/ScreenHeight>
   <ScreenWidth>640<
/ScreenWidth>
   <Timezone>CET</Timezone>
   <UserAgent>Mozilla/4.0 (MS
IE 6.0; Windows NT 5.0)<
/UserAgent>
   <JavaEnabled>true<
/JavaEnabled>
   <ScreenColorDepth>8<
/ScreenColorDepth>
   <ChallengeWindow>5<
/ChallengeWindow>
</Browser></pre></td><td>HTTP accept header sent from the cardholder's browser.</td><td>Length: Variable, maximum 2048 characters JSON Data Type: String Value accepted: If the total length of the accept header sent by the browser exceeds 2048 characters, the 3DS Server truncates</td></tr>
<tr><td>Language</td><td>The cardholder's browser language.</td><td>Length: Variable, 1–8 characters JSON Data Type: String</td></tr>
<tr><td>Screen height</td><td>This field contains the total height of the cardholder's screen in pixels.</td><td>Length: Variable, 1–6 characters JSON Data Type: String</td></tr>
<tr><td>Screen width</td><td>This field contains the total width of the cardholder's screen in pixels.</td><td>Length: Variable, 1–6 characters JSON Data Type: String</td></tr>
<tr><td>Browser timezone</td><td>This field contains the cardholder's browser local timezone.</td><td>Length: 1–5 characters JSON Data Type: String Value accepted: Value is returned from the getTimezoneOffset() method.</td></tr>
<tr><td>User agent</td><td>This field contains the exact content of the HTTP User-Agent header.</td><td>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.</td></tr>
<tr><td>Java enabled</td><td>true/false - Ability of the cardholder's browser to execute Java.</td><td>JSON Data Type: Boolean Values accepted: • true • false</td></tr>
<tr><td>Screen color depth</td><td>This field contains a value representing the bit depth of the color palette, in bits per pixel, for displaying images.</td><td>Length: 1–2 characters JSON Data Type: String Values accepted: 1 = 1 bit • 4 = 4 bits • 8 = 8 bits • 15 = 15 bits • 16 = 16 bits • 24 = 24 bits • 32 = 32 bits • 48 = 48 bits</td></tr>
<tr><td>Authentication window size</td><td>Size of the authentication iframe which will render the ACS authentication front-end to the shopper for interaction.

Please send an Integer between 1-5. The integer corresponds to one the following resolutions:
<table><tr><td>1</td><td>2</td><td>3</td><td>4</td><td>5</td></tr><tr><td>250 x 400</td><td>390 x 400</td><td>500 x 60</td><td>600 x 400</td><td>Full screen</td></tr></table></td><td></td></tr>
<tr><td>IP address</td><td><pre><Contact>
    <Ip>101.202.011.022</Ip>
</Contact></pre></td><td>IP address of the cardholder's browser</td><td>Length: Variable, maximum 45 characters JSON Data Type: String Value accepted:

IPv4 address is represented in the dotted decimal format of 4 sets of decimal numbers separated by dots. The decimal number in each and every set is in the range 0 to 255. Example IPv4 address: 1.12.123.255

IPv6 address is represented as eight groups of four hexadecimal digits, each group representing 16 bits (two octets. The groups are separated by colons (:). Example IPv6 address:2011:0db8:85 a3: 0101:0101:8a2e:03 70:7334</td></tr>
</table>

Server to Server request

```
<?xml version="1.0" encoding="UTF-8"?>
<Request version="1.0">
   <Header>
      <Security sender="8426fe6246d5cd69c28d1350324ea040" />
```

```
   </Header>
   <Transaction channel="8426fe6246d5cd69c28d1350324ea040" mode="
CONNECTOR_TEST" response="ASYNC" source="XML">
      <User login="37901b7c3441a4f0bad4cab062d0ed19" pwd="123123" />
      <Payment code="CC.DB">
         <Presentation>
            <Amount>92.00</Amount>
            <Currency>EUR</Currency>
         </Presentation>
      </Payment>
      <Account>
         <Number>4111111111111111</Number>
         <Holder>Test Tester</Holder>
         <Brand>VISA</Brand>
         <Year>2019</Year>
         <Month>10</Month>
         <Verification>123</Verification>
         <Expiry month="10" year="2019" />
      </Account>
      <Customer>
         <Name>
            <Given>Joe</Given>
            <Family>Doe</Family>
         </Name>
         <Address>
            <Street>Leopoldstr. 1</Street>
            <Zip>80798</Zip>
            <City>München</City>
            <State>BY</State>
            <Country>DE</Country>
         </Address>
         <Contact>
            <Email>test.test@mail.com</Email>
            <Ip>123.123.123.12</Ip>
            <Phone>+49 179 520 2990</Phone>
         </Contact>
      </Customer>
      <Frontend>
         <ResponseUrl>https://testRedirect.merchant.com</ResponseUrl>
      </Frontend>
      <Browser>
         <AcceptHeader>text/html,application/xhtml+xml,application/xml;q=0.
9,image/webp,image/apng,*/*;q=0.8,application/signed-exchange;v=b3<
/AcceptHeader>
         <Language>en-US</Language>
         <ScreenHeight>200</ScreenHeight>
         <ScreenWidth>400</ScreenWidth>
         <Timezone>CET</Timezone>
         <UserAgent>Mozilla/5.0 (X11; Linux x86_64) AppleWebKit/537.36
(KHTML, like Gecko) Chrome/73.0.3683.103 Safari/537.36</UserAgent>
         <JavaEnabled>false</JavaEnabled>
         <ScreenColorDepth>24</ScreenColorDepth>
         <ChallengeWindow>5</ChallengeWindow>
```

```
        </Browser>
     </Transaction>
  </Request>
```

Server to Server response

```
<?xml version="1.0" encoding="UTF-8"?>
<Response version="1.0">
   <Transaction mode="CONNECTOR_TEST" channel="8a829417291263d5012920bc6fec0390" response="ASYNC">
      <Identification>
         <ShortID>5479.2516.4706</ShortID>
         <UniqueID>8a82944a4ba70955014bbf94127d2831</UniqueID>
         <TransactionID>201502240933156f181be0695eba4be0890f2dd49fe18f</TransactionID>
         <ShopperID>shopper123</ShopperID>
      </Identification>
      <Payment code="CC.DB" />

<Authentication type="3DSecure">
  <3DSecureVersion>1.0 | 2.0 | 2.1 | 2.2</3DSecureVersion>
  <3DSTransactionId>txid</DSTransactionId>
  <3DSecureFlow>challenge | frictionless</3DSecureFlow>
</Authentication>

      <Processing code="CC.DB.80.00">
         <Timestamp>2015-02-25 07:12:07</Timestamp>
         <Result>ACK</Result>
         <Status code="80">WAITING</Status>
         <Reason code="00">Transaction Pending</Reason>
         <Return code="000.200.000">Transaction pending</Return>
         <Redirect url="https://test.ppipe.net/connectors/demo/simulator.link?REMOTEADDRESS">
            <Parameter name="connector">THREEDSECURE</Parameter>
            <Parameter name="MD">8a82944a4ba70955014bbf9412f72839</Parameter>
            <Parameter name="TermUrl">https://test.ppipe.net/connectors9</Parameter>
            <Parameter name="PaReq">IT8ubu+5z4YupUCOEHKsbiPep8UzIAcPKJEjpwGlzD8#KioqKioqKioqKioqMTExMSM
yLjUwIEVVUiM</Parameter>
         </Redirect>
         <Risk score="100" />
         <ConnectorDetails>
            <Result name="redirect.preconditions[0].origin">iframe#hidden</Result>
            <Result name="redirect.preconditions[0].waitUntil">iframe#load</Result>
            <Result name="redirect.preconditions[0].description">Hidden iframe post for 3D Secure 2.0<
/Result>
            <Result name="redirect.preconditions[0].method">POST</Result>
            <Result name="redirect.preconditions[0].url">methodUrl</Result>
            <Result name="redirect.preconditions[0].parameters[0].name">methodData</Result>
            <Result name="redirect.preconditions[0].parameters[0].value">methodData</Result>
         </ConnectorDetails>
      </Processing>

   </Transaction>
</Response>
```

How to handle the methodUrl and methodData

For OPP Server-to-Server and XML Server-to-Server, the handling of the methodUrl and methodData must be done by the integrator. Integrators using OPP COPYandPAY will benefit from the COPYandPAY in-build javascript engine which will handle the methodUrl and methodData automatically.

Following steps have to be executed for OPP Server-to-Server and XML Server-to-Server after receiving the the gateway response on the initial request.

1. Open a hidden iframe and post data to the methodURL

```
<form name='' action='call.url' method='POST'>
        <INPUT type='hidden' name='call.parameters[].name'
value='call.parameters[].value'>
</form>
<script>
        window.onload = submitForm;
        function submitForm() { downloadForm.submit(); }
</script>
```

2. Redirect the shopper within and iframe to the redirect URL if onLoad event received from 1.

```
<form name='' action='redirect.URL' method='POST'>
        <INPUT type='hidden' name='redirect.parameters[].name'
value='redirect.parameters[].value'>
</form>
<script>
        window.onload = submitForm;
        function submitForm() { downloadForm.submit(); }
</script>
```

Fields required for 3D Secure 2.0

Please note that in order to have a better rate of successful risk-checks during the risk based authentication, it is recommended to send as many fields as possible. This will positively affect the number of frictionless flows.

Source is the cardholder or cardholder's environment

Field name	Mandatory/Optional	Source when using the CopyAndPay widget	Source when integrating via Server-to-Server API
card.expiryMonth	Mandatory	Cardholder (via widget)	Cardholder (via API)
card.expiryYear	Mandatory	Cardholder (via widget)	Cardholder (via API)
card.number	Mandatory	Cardholder (via widget)	Cardholder (via API)
billing.city	Required unless market or regional mandate restricts sending this information.	Cardholder (via widget or API)	Cardholder (via API)
billing.country	Required unless market or regional mandate restricts sending this information.	Cardholder (via widget or API)	Cardholder (via API)
billing.street1	Required unless market or regional mandate restricts sending this information.	Cardholder (via widget or API)	Cardholder (via API)
billing.postcode	Required unless market or regional mandate restricts sending this information.	Cardholder (via widget or API)	Cardholder (via API)
customer.email	Required unless market or regional mandate restricts sending this information.	Cardholder (via widget or API)	Cardholder (via API)
customer.givenName	Required unless market or regional mandate restricts sending this information.	Cardholder (via widget or API)	Cardholder (via API)

customer.surname	Required unless market or regional mandate restricts sending this information.	Cardholder (via widget or API)	Cardholder (via API)
amount	Mandatory	Payment (via API)	Payment (via API)
currency	Mandatory	Payment (via API)	Payment (via API)
shipping.city	Optional	Cardholder (via widget or API)	Cardholder (via API)
shipping.country	Optional	Cardholder (via widget or API)	Cardholder (via API)
shipping.street1	Optional	Cardholder (via widget or API)	Cardholder (via API)
shipping.street2	Optional	Cardholder (via widget or API)	Cardholder (via API)
shipping.postcode	Optional	Cardholder (via widget or API)	Cardholder (via API)
shipping.state	Optional	Cardholder (via widget or API)	Cardholder (via API)
billing.street2	Optional	Cardholder (via widget or API)	Cardholder (via API)
billing.street2	Optional	Cardholder (via widget or API)	Cardholder (via API)
billing.state	Optional	Cardholder (via widget or API)	Cardholder (via API)
customer.phone	Optional	Cardholder (via widget or API)	Cardholder (via API)
customer.workPhone	Optional	Cardholder (via widget or API)	Cardholder (via API)
customer.mobile	Optional	Cardholder (via widget or API)	Cardholder (via API)
customer.browser.acceptHeader	Mandatory	Automatically collected by the widget	Merchant should collect and send via API
customer.browser.language	Mandatory	Automatically collected by the widget	Merchant should collect and send via API
customer.browser.screenHeight	Mandatory	Automatically collected by the widget	Merchant should collect and send via API
customer.browser.screenWidth	Mandatory	Automatically collected by the widget	Merchant should collect and send via API
customer.browser.timezone	Mandatory	Automatically collected by the widget	Merchant should collect and send via API
customer.browser.userAgent	Mandatory	Automatically collected by the widget	Merchant should collect and send via API
customer.browser.ipAddress	Optional	Automatically collected by the widget	Merchant should collect and send via API
customer.browser.javaEnabled	Optional	Automatically collected by the widget	Merchant should collect and send via API
customer.browser.screenColorDepth	Optional	Automatically collected by the widget	Merchant should collect and send via API
customer.browser.challengeWindow	Optional	Automatically collected by the widget	Merchant should collect and send via API

Source is the merchant

Field name	Mandatory/Optional	Comment
Merchant category code	Mandatory	
Merchant country code	Mandatory	
Merchant name	Mandatory	Merchant name assigned by the Acquirer or Payment System.
Merchant ID	Mandatory	Acquirer-assigned Merchant identifier.
Requestor ID	Mandatory	DS assigned 3D Secure Requestor identifier.
Requestor Name	Mandatory	DS assigned 3D Secure Requestor name.

Requestor URL	Mandatory	Fully qualified URL of 3D Secure Requestor website or customer care site.

Optional settings

Merchants have the possibility to set the preference of a transaction being challenged or not. This really is only a preference, and won't guarantee that the issuer will or will not request a challenge from the cardholder. It is up to the issuer if they consider the merchant's preference, and if they include it when they assess the risk of the transaction.

For example when a card is being stored for later use (eg. for One click checkout), a challenge may be requested by the merchant. In another example, there might be some regional mandates that certain transactions have to be challenged and the merchant should ask for a mandated challenge.

Send the field threeDSecure.challengePreference with one of the following values:

Value	Challenge Preference	Description
01	No preference	The merchant has no preference, and fully trust the issuer to ask a challenge from the cardholder.
02	No challenge requested	The merchant prefers that the cardholder is not authenticated by the issuer, and only the frictionless flow applies
03	Challenge requested: 3D Secure Requestor Preference	The merchant prefers that the cardholder is authenticated by the issuer.
04	Challenge requested: Mandate	The cardholder authentication is mandated (eg. by regional mandates)

The field threeDSecure.challengePreference is optional. If not sent, the value "01 - No preference" applies by default.

Information about the cardholder's account and history with the merchant

The following fields are not mandatory, but it is strongly recommended to send them. They are affecting the accuracy of the issuer's risk check, and will result in more frictionless flows.

The field values below can be collected by the 3D Secure Requestor* about the cardholders activity on their webshop.

**3D Secure Requestor denotes the merchant*

Field name	Description

<table>
<tr><td>customParameters[ReqAuthMethod]</td><td>Method used by the Cardholder to authenticate to the 3D Secure Requestor.

Contains optional information about how the cardholder authenticated during login to their 3D Secure Requestor account.

Possible values are:

<table><tr><td>01</td><td>No authentication occurred (i. e. cardholder "logged in" as guest)</td></tr><tr><td>02</td><td>Login to the cardholder account at the merchant system using cardholder's own credentials</td></tr><tr><td>03</td><td>Login to the cardholder account at the merchant system using federated ID</td></tr><tr><td>04</td><td>Login to the cardholder account at the merchant system using issuer credentials</td></tr><tr><td>05</td><td>Login to the cardholder account at the merchant system using third-party authentication</td></tr><tr><td>06</td><td>Login to the cardholder account at the merchant system using FIDO Authenticator</td></tr></table></td></tr>
<tr><td>customParameter[ReqAuthTimestamp]</td><td>Date and time in UTC of the cardholder authentication. Accepted date format is YYYYMMDDHHMM.

Part of the 3D Secure Requestor Authentication Information which contains optional information about how the cardholder authenticated during login to their account.</td></tr>
<tr><td>customParameter[PriorAuthMethod]</td><td>Mechanism used by the Cardholder to previously authenticate to the 3D Secure Requestor.

Contains information about a 3D Secure cardholder authentication that occurred prior to the current transaction.

Possible values are:

<table><tr><td>01</td><td>Frictionless authentication occurred by ACS</td></tr><tr><td>02</td><td>Cardholder challenge occurred by ACS</td></tr><tr><td>03</td><td>ACS verified</td></tr><tr><td>04</td><td>Other issuer methods</td></tr></table></td></tr>
<tr><td>customParameters[PriorAuthTimestamp]</td><td>Date and time in UTC of the prior cardholder authentication. Accepted date format is YYYYMMDDHHMM.

Contains information about a 3D Secure cardholder authentication that occurred prior to the current transaction.</td></tr>
</table>

<table>
<tr><td>customParameter[PriorReference]</td><td>This data element provides additional information to the ACS to determine the best approach for handling a request. It contains an ACS Transaction ID for a prior authenticated transaction (for example, the first recurring transaction that was authenticated with the cardholder).

Contains information about a 3D Secure cardholder authentication that occurred prior to the current transaction.</td></tr>
<tr><td>customParameter[AccountId]</td><td>Additional information about the account optionally provided by the 3D Secure Requestor in AReq messages.</td></tr>
<tr><td>customParameter[AccountAgeIndicator]</td><td>Length of time that the cardholder has had the account with the 3D Secure Requestor.

Possible values are:

<table><tr><td>01</td><td>No account (guest check-out)</td></tr><tr><td>02</td><td>Created during this transaction</td></tr><tr><td>03</td><td>Less than 30 days</td></tr><tr><td>04</td><td>30-60 days</td></tr><tr><td>05</td><td>More than 60 days</td></tr></table></td></tr>
<tr><td>customParameter[AccountChangeDate]</td><td>Date that the cardholder's account with the 3D Secure Requestor was last changed. Accepted date format is YYYYMMDD.</td></tr>
<tr><td>customParameter[AccountChangeIndicator]</td><td>Length of time since the cardholder's account information with the 3D Secure Requestor was last changed.

Possible values are:

<table><tr><td>01</td><td>No account (guest check-out)</td></tr><tr><td>02</td><td>Created during this transaction</td></tr><tr><td>03</td><td>Less than 30 days</td></tr><tr><td>04</td><td>30-60 days</td></tr><tr><td>05</td><td>More than 60 days</td></tr></table></td></tr>
<tr><td>customParameter[AccountDate]</td><td>Date that the cardholder opened the account with the 3D Secure Requestor. Accepted date format is YYYYMMDD.</td></tr>
<tr><td>customParameter[AccountPasswordChangeDate]</td><td>Date that cardholder's account with the 3D Secure Requestor had a password change or account reset. Accepted date format is YYYYMMDD.</td></tr>
<tr><td>customParameter[AccountPasswordChangeIndicator]</td><td>Indicates the length of time since the cardholder's account with the 3D Secure Requestor had a password change or account reset. Possible values are:

<table><tr><td>01</td><td>No account (guest check-out)</td></tr><tr><td>02</td><td>Created during this transaction</td></tr><tr><td>03</td><td>Less than 30 days</td></tr><tr><td>04</td><td>30-60 days</td></tr><tr><td>05</td><td>More than 60 days</td></tr></table></td></tr>
<tr><td>customParameter[AccountPurchaseCount]</td><td>Number of purchases with this cardholder account during the previous six months.</td></tr>
<tr><td>customParameter[AccountProvisioningAttempts]</td><td>Number of Add Card attempts for the account in the last 24 hours.</td></tr>
</table>

<table>
<tr><td>customParameter[AccountDayTransactions]</td><td>Number of transactions (successful and abandoned) for this cardholder account with the 3D Secure Requestor across all payment accounts in the previous 24 hours.</td></tr>
<tr><td>customParameter[AccountYearTransactions]</td><td>Number of transactions (successful and abandoned) for this cardholder account with the 3D Secure Requestor across all payment accounts in the previous year.</td></tr>
<tr><td>customParameter[PaymentAccountAge]</td><td>Date that the payment account was enrolled in the cardholder's account with the 3D Secure Requestor. Accepted date format is YYYYMMDD.</td></tr>
<tr><td>customParameter[PaymentAccountAgeIndicator]</td><td>Indicates the length of time that the payment account was enrolled in the cardholder's account with the 3D Secure Requestor.

Possible values are:<table><tr><td>01</td><td>No account (guest check-out)</td></tr><tr><td>02</td><td>Created during this transaction</td></tr><tr><td>03</td><td>Less than 30 days</td></tr><tr><td>04</td><td>30-60 days</td></tr><tr><td>05</td><td>More than 60 days</td></tr></table></td></tr>
<tr><td>customParameter[ShipAddressUsageDate]</td><td>Date when the shipping address used for this transaction was first used with the 3D Secure Requestor. Accepted date format is YYYYMMDD.</td></tr>
<tr><td>customParameter[ShipAddressUsageIndicator]</td><td>Indicates the length of time since the shipping address used for this transaction was first used with the 3D Secure Requestor.

Possible values are:<table><tr><td>01</td><td>No account (guest check-out)</td></tr><tr><td>02</td><td>Created during this transaction</td></tr><tr><td>03</td><td>Less than 30 days</td></tr><tr><td>04</td><td>30-60 days</td></tr><tr><td>05</td><td>More than 60 days</td></tr></table></td></tr>
</table>

<table>
<tr>
<td>customParameter[ShipIndicator]</td>
<td>Indicates shipping method chosen for the transaction. Merchants must choose the Shipping Indicator code that most accurately describes the cardholder's specific transaction, not their general business. If one or more items are included in the sale, the Shipping Indicator code for the physical goods is used, or if all digital goods, the Shipping Indicator code that describes the most expensive item.

Possible values are:

<table>
<tr><td>01</td><td>Ship to cardholder's billing address</td></tr>
<tr><td>02</td><td>Ship to another verified address on file with merchant</td></tr>
<tr><td>03</td><td>Ship to address that is different than the cardholder's billing address</td></tr>
<tr><td>04</td><td>"Ship to Store" / Pick-up at local store (Store address shall be populated in shipping address fields)</td></tr>
<tr><td>05</td><td>Digital goods (includes online services, electronic gift cards and redemption codes)</td></tr>
<tr><td>06</td><td>Travel and Event tickets, not shipped</td></tr>
<tr><td>07</td><td>Other (for example, Gaming, digital services not shipped, emedia subscriptions, etc.)</td></tr>
</table>
</td>
</tr>
<tr>
<td>customParameter[ShipNameIndicator]</td>
<td>Indicates if the Cardholder Name on the account is identical to the shipping Name used for this transaction.

Possible values are:

<table>
<tr><td>01</td><td>Account Name identical to shipping Name</td></tr>
<tr><td>02</td><td>Account Name different than shipping Name</td></tr>
</table>
</td>
</tr>
<tr>
<td>customParameter[SuspiciousAccountActivity]</td>
<td>Indicates whether the 3D Secure Requestor has experienced suspicious activity (including previous fraud) on the cardholder account.

Possible values are:

<table>
<tr><td>01</td><td>No suspicious activity has been observed</td></tr>
<tr><td>02</td><td>Suspicious activity has been observed</td></tr>
</table>
</td>
</tr>
</table>