

DP - 3DSecure user in direct mode V1

Content



[Introduction](#)
[3D-Secure in Direct Interface mode with a payment](#)
[Step 1 - verifyEnrollment](#)
[Step 2 : doAuthorization with 3D Secure settings](#)
[Back Office](#)
[3D Secure payment scheme](#)

Introduction

What is 3DSecure ?

The 3DSecure process adds a security layer to an online purchase: the buyer will usually validate his payment by entering a one-time passcode provided by their bank via text message. Other methods also exist. When a transaction is authenticated with 3DSecure, the liability in case of a subsequent chargeback is borne by the issuing bank. This is referred to as the principle of liability shift.

Payline holds Visa and Mastercard International (MCI) 3DSecure certifications.

Subscription

The merchant gets a 3DSecure Distance Selling contract with their bank and sends the details to Payline.

Payline registers the 3DSecure contract with Visa and MCI then activates the contracts. Allow up to ten working days for activation.

Prerequisites for using Payline payment solution

The 3DSecure solution in Direct interface mode ensures the secure transfer of sensitive data and processes authentication and authorization requests.

Integration points :

- [verifyEnrollment](#) is required to provide authentication and [doAuthorization](#) to perform the authorization;
- get the result of the transaction with [getTransactionDetails](#).

You must check the [service access key](#) and configure the [SOAP UI](#).

3D-Secure in Direct Interface mode with a payment

The following steps present [verifyEnrollment](#) and [doAuthorization](#) web services for realizing 3DSecure transactions using the Direct interface.

Step 1 - verifyEnrollment

This first call is to verify the eligibility of the card to a 3DSecure authentication, and therefore to know if the cardholder is registered with a VISA or Mastercard Directory Server. This is done with the verifyEnrollment web service.

Example of a request and response for the verifyEnrollment web service :

verifyEnrollmentRequest (VEReq)	verifyEnrollmentResponse (VERes)
--	---

<pre> <impl:verifyEnrollmentRequest> <impl:card> <obj:number>4970100000325734</obj: number> <obj:type>CB</obj:type> <obj:expirationDate>0912</obj: expirationDate> <obj:cvx>123</obj:cvx> </impl:card> <impl:payment> <obj:amount>4050</obj:amount> <obj:currency>978</obj:currency> <obj:action>100</obj:action> <obj:mode>CPT</obj:mode> <obj:contractNumber>CB3DS</obj: contractNumber> </impl:payment> <impl:orderRef>REF0923847</impl:orderRef> </impl:verifyEnrollmentRequest> </pre>	<pre> <verifyEnrollmentResponse> <result> <code>03000</code> <shortMessage>ACCEPTED</shortMessage> <longMessage>Operation Successful</longMessage> </result> <actionUrl>https://acs.modirum.com/mdpayacs/pareq</actionUrl> <actionMethod>POST< /actionMethod> <pareqFieldName>PaReq</pareqFieldName> <pareqFieldValue> eJxVkdтуwJAMhl+I4gGaA21ZkcNEOGhIYzAYQ9rNFFoPKq CIScvh7ZeUMrbcxJ9jx/ZveN8oxP4co1KhgDFqLdfoJHGnw XgrpM2QNQRMuzPMBRrX6SRLBXOpy4Hc0GSpaCPTQoCM 8qfRq2C86fkBkBphj2rUF+x6gFwRURlH0e1NnRiPQCqCKCv TQl0E9ymQG0CpdmJTFic2lafTyV1n2XqH7rciGqUp/Zh3TM TXKiuLJC9RA7EJQO59TUtraVPgnMRivPgMJkt/uNoO5Xzrl 5OBz5eDj5fZ8K0DxEZALasUnJp2KQ8cGrY5a3tmosoPcm8 7E4PFzPGoa1utPXCwhbpX8Kh9+esBo7LCNLqIsPVg5rsR4 PmQpWgijKy/NsSolzNGfd1n6D1bpaPCiNiklIdBJXXF9qfES MY4DauvLACxGaTelqmXbKx/y/8Ba4usNQ== </pareqFieldValue> <termUrlName>TermUrl</termUrlName> <termUrlValue> https://acs.modirum.com/mdpayacs.php </termUrlValue> <mdFieldName>MD</mdFieldName> <mdFieldValue>1Fz9nEnAZJNn8NvXEKDT</mdFieldValue> </verifyEnrollmentResponse> </pre>
---	--

Once enrolment has been confirmed, an authentication call to the ACS server can be initiated. Information received from the verifyEnrollmentResponse must be sent to the authentication server.

Sending information

To send this information :

- in POST : create an HTML form,
- in GET : create a link.

POST : The data information will be sent to the authentication server via the form below.
The field names and values are dynamically retrieved from the verifyEnrollmentResponse :

- Payment session:
 - mdFieldName = **MD**
 - mdFieldValue = **1Fz9nEnAZJNn8NvXEKDT**
- Authentication request:
 - pareqFieldName = **PaReq**
 - pareqFieldValue = **eJxVkdтуwJAMhl+I4gGaA...**
- Authentication adress server. This address must retrieve a form sent in POST.
 - termUrlName = **TermUrl**
 - termUrlValue = **https://acs.modirum.com/mdpayacs.php**

Sample HTML form to perform a test on your server:

HTML form
<pre> <form name="downloadForm" action="https://acs.modirum.com/mdpayacs/pareq" method="POST"> <input type="hidden" name="TermUrl" value="http://127.0.0.1/3DSecure/receive_form.php"> PAREQ : <input type="text" name="PaReq">
 MD : <input type="text" name="MD">
 <input type="submit" name="submit" value="Submit"> </form> </pre>

Receipt of information returned during authentication

The authentication server sends its message to the URL entered in the TermURL parameter (sent in the previous form). In the response form, two fields must be retrieved to continue the transaction in 3DSecure mode:

- The MD field: always the same field allowing the follow-up of the session
- the Payer Authentication Response (PaRes) field: an encrypted string containing the response of the authentication server. The value of the PaRes field will validate or not the transaction as a 3D Secure transaction.

These two fields are retrieved and allow to complete the doAuthorizationRequest in 3D Secure mode.

Sample script (here written in PHP) to retrieve the response to authentication :

Script PHP : receive_form.php
<pre><?php \$pares = \$_POST['PaRes']; \$md = \$_POST['MD']; echo "MD : ".\$md."
PARES : ".\$pares; ?></pre>

Note: This script must be placed on a started web server and in a folder corresponding to the address sent via the TermURL field.

Example: if the server is local it is quite possible to put as value:

TermURL = http://127.0.0.1/3DSecure/receive_form.php

Step 2 : doAuthorizathion with3D Secure settings

The doAuthorization service allows you to perform the transaction with the 3D Secure parameters.

The parameters provided : md / pares permit to check user authentication and thus the user identity before carrying out the transaction.

If the parameters are correct, the transaction is carried out as authorization request.

doAuthorizationRequest	doAuthorizationResponse
------------------------	-------------------------

<pre> <impl:doAuthorizationRequest> <impl:payment> <obj:amount>4150</obj:amount> <obj:currency>978</obj:currency> <obj:action>100</obj:action> <obj:mode>CPT</obj:mode> <obj:contractNumber>CB3DS</obj:contractNumber> </impl:payment> <impl:card> <obj:number>4970105512345674</obj:number> <obj:type>CB</obj:type> <obj:expirationDate>0912</obj:expirationDate> <obj:cvx>123</obj:cvx> </impl:card> <impl:order> <obj:ref>REF023493</obj:ref> <obj:country>FR</obj:country> <obj:taxes>100</obj:taxes> <obj:amount>1400</obj:amount> <obj:currency>978</obj:currency> <obj:date>28/01/2009 09:32</obj:date> </impl:order> <impl:buyer> <obj:lastName>Dupond</obj:lastName> <obj:firstName>Wilfried</obj:firstName> <obj:email>wilfried.dupond@yahoo.fr</obj:email> </impl:buyer> <impl:authentication3DSecure> <obj:md>xRtMifcy975D2EB3Zs8e</obj:md> <obj:pares> eJzFV2mTokoW/Ssd/T4a3ewKHZQq8LT8uWh9v0X8C9X 9dnSvZpwiZxtkQnR4/vcxQo0vM1a4/II9R/BFjkeQryXL4 NU12Tb4MZVE1L1+PbVv/QJC+77/3xPfzNUWmgFEEZZ k6R9fX0cle6U6nJcsH1bnKovDiruH7bTYMGmP5/2X9wl 2H14xxBT5b5PbbzFGVt8eCEo8aYT83umHcP/OLJ8Dvzb YYYo8JPJlasmZySB7LnHxxTOXI6x8fSC1kadK0/86Mb7N Dmzw2LW7JsXdOgDbKqGt0MWzXUzHgfeTiJHYyXt3Gvli LP+N9W4D2XV0MrlQkUn+/iOLJrhOdX5t6je0MVLvrO6/ +UWYynOS9H7sYGAZ5U3lbnDcT3ZMMEcjDfJb20VXhTw bWgWEot2lx04i1tmBAuFHx2aEgzgEtcaJzH8TLbsXbpj4r </obj:pares> <obj:xid/> <obj:eci/> <obj:cavv/> <obj:cavvAlgorithm/> <obj:vadsResult/> </impl:authentication3DSecure> </impl:doAuthorizationRequest> </pre>	<pre> <doAuthorizationResponse> <result> <code>00000</code> <shortMessage>ACCEPTED</shortMessage> <longMessage>Transaction approved< /longMessage> </result> <transaction> <id>90217095220928</id> <date>17/02/09 09:52</date> <isDuplicated>0</isDuplicated> <isPossibleFraud>0</isPossibleFraud> <fraudResult/> <explanation/> <threeDSecure>Y</threeDSecure> <score/> </transaction> <authorization> <number>A55A</number> <date>17/02/09 09:52</date> </authorization> </doAuthorizationResponse> </pre>
--	--

Back Office

Menu 'Technical follow-up of webservice calls' to find the call of the web service verifyEnrollment allows to see the details of the verifyEnrollment.

The result of the 3DSecure transaction is then visible in the Payline Administration Center: on the results of a search and in the detail of the transaction 3DSecure tab.

Screen searches for transactions:

Transactions search											
Search Results											
Criteria reminder Show all transactions - Merchant ID : 28560213285346 (DEMO_PAYLINE) - Transactions accepted - 3D Secure card : ALL Total: 236 transactions											
ID	Ref cmd	Date trans	Amount	Transaction Type	Back	MoP	Point of sale	Payment data	Customer name	Email address	
G1903140843391123	PHP1552549418	03/14/2019 08:44:13	EUR300.00	Authorization+Validation	00000 ✓	KLARNA_PAYNOW	Demo Payline	DE86000000XXXXXXX02	hervé dupont	h.d@yopmail.com	
G190311151513431	PHP1552313718	03/11/2019 15:16:11	EUR300.00	Authorization+Validation	00000 ✓	KLARNA_PAYNOW	Demo Payline	DE30000000XXXXXXX99	hervé dupont	h.d@yopmail.com	
G1903111512433411	PHP1552313586	03/11/2019 15:13:53	EUR1.00	Authorization+Validation	00000 ✓	TICKET_PREMIUM	Demo Payline		Iza BELLE	iza.belle@yopmail.com	
19066161945904	PHP1551971989	03/07/2019 16:19:45	EUR300.00	Authorization+Validation	00000 ✓	CB	Demo Payline	411111XXXXXX1111	hervé dupont	h.d@yopmail.com	
G1903041155152483	NR_ONEY_IT_3x001_TC2_20190304-1154	03/04/2019 11:57:17	EUR172.04	Authorization+Validation	00000 ✓	ONEY	Demo Payline		Rossella Mirti	maJJWvKokslmUWw@	
G1903012231239223	PHP1551475886	03/01/2019 22:31:51	EUR1.00	Authorization+Validation	00000 ✓	TICKET_PREMIUM	Demo Payline		John DOE	johndoe@yopmail.com	
G1903012231449241	PHP1551475886	03/01/2019 22:31:44	EUR1.00	Authorization+Validation	00000 ✓	TICKET_PREMIUM	Demo Payline		John DOE	johndoe@yopmail.com	
G1903012231268643	PHP1551475289	03/01/2019 22:23:10	EUR1.00	Authorization+Validation	00000 ✓	TICKET_PREMIUM	Demo Payline		John DOE	johndoe@yopmail.com	
G1903012232038741	PHP1551475289	03/01/2019 22:23:03	EUR1.00	Authorization+Validation	00000 ✓	TICKET_PREMIUM	Demo Payline		John DOE	johndoe@yopmail.com	

3D Secure transaction Details:

3DSECURE AND PAYMENT GUARANTEE

Merchant name displayed **DEMO_PAYLINE**
Transfert of responsibility **No (Transaction ineligible)**
Enrolled **Yes (Y)**
Authenticated **Yes (Y)**
md **1zCRYcxWF8cbpv1Ny5zi**
xid **MXpDULLjeFdGOGNICHYxTnk1emk=**
cryptogram **AAABCCIBIQAAAAIUKEhAAAAA=**
algorithm of the cryptogram **2**
eci **05**
effectiveAuthType
Merchant Challenge Ind
Trans Status
Trans Status Reason
ChallengeCancelInd
Scheme Score
Ds Trans ID

3D Secure payment scheme

- The consumer validates his cart shopping then the merchant prepares web page to where will be filled the payment data.
A VEReq (Verification Enrollment Request) message allows access to Directory Server to verify card registration in the directory containing cards declared "enlisted" 3-D Secure and provide ACS URL.
Verification enrollment response containing authentication result, that will be returned to Merchant Plug-in (MPI) to manage the dialogue with Directory and ACS to allow the buyer to authenticate.
- The merchant redirects the consumer to ACS URL for authentication.
The request "PAREq" (Payer authentication request) allows access to bank ACS to trigger the authentication phase.
The response "PAREs" (Pay authentication response), containing the authentication result of cardholder will be transmitted to the merchant.
- The merchant can trigger a request for authorization and payment validation by calling service doAuthorizationRequest.
- The merchant retrieves details transaction by calling service getTransactionDetails.

