Moneris MPI – Verified by Visa

COM API –v1.01

Table of Contents

<u>** PLEASE READ CAREFULLY****</u>**

You have a responsibility to send only Visa cards to the VBV MPI. Under no circumstances should ANY other card type be sent to the VBV MPI.

About this Documentation

This documentation contains the basic information for using the JAVA API for sending a Verified by Visa request to the Moneris MPI. In particular it describes the format for sending the requests and handling the corresponding responses you will receive.

1. System and Skill Requirements

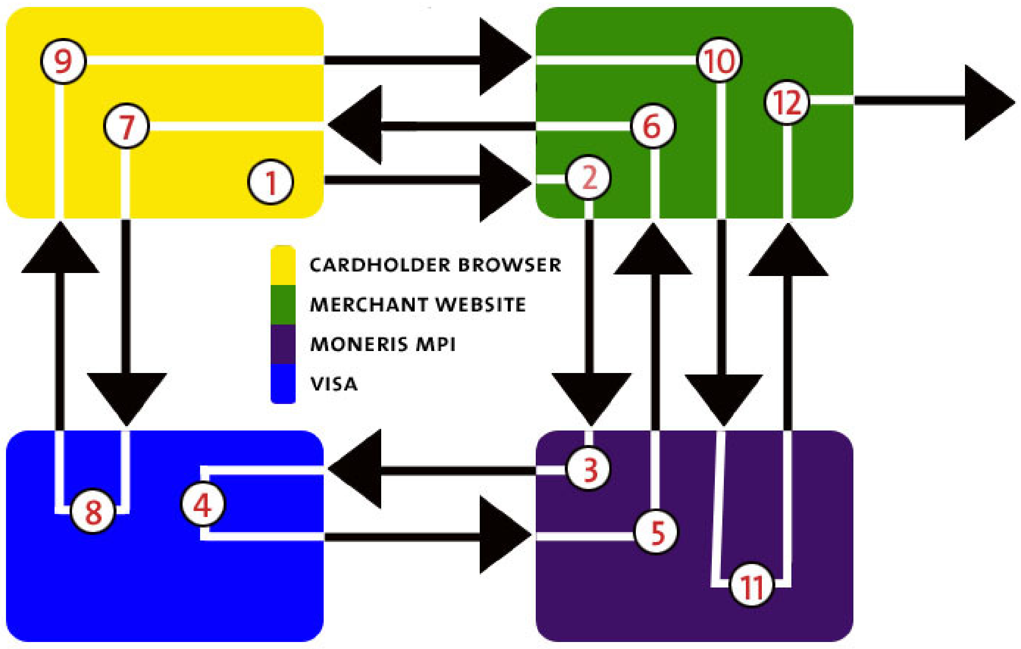
In order to use the COM API your system will need to have the following:
1. SSL Certificate
2. Port 43924 open for bi directional communication
3. WinHTTP 5.1 (Please not that this will not work with winHTTP 5.0)

note: Core SDK is no longer required.

As well, you will need to have the following knowledge and/or skill set:
1. Permissions to register a dll
2. Knowledge of a COM compatible language (ASP or VBscript)

2. Transaction Flow

Below is a diagram with explanations about the flow of a VBV transaction.



1. Cardholder enters their credit card number and submits their transaction information to the merchant.

2. Upon receiving the transaction request the merchant calls the Moneris MPI API and passes a TXN type request.

3. The Moneris MPI receives the request and authenticates the merchant and send the transaction information to Visa.

4. Visa verifies if the card is enrolled and returns a URL for the issuer.

5. Moneris MPI receives the response from Visa and forwards the information to the merchant.

6. The Moneris MPI API installed at the merchant receives the response from the Moneris MPI. If the card is enrolled the merchant makes a call to the API which opens a popup on the Cardholder browser. *If the card is not enrolled a transaction can be sent to the processor identifying it as VBV attempted.

7. The cardholder browser uses the URL returned from visa via the merchant to communicate directly to the bank. The contents of the popup are loaded and the cardholder enters their pin.

8. The information is submitted to the bank and authenticated. A response is then returned to the client browser.

9. The client browser receives the response from the bank and forwards it to the merchant.

10. The merchant receives the response information from the cardholder browser and makes a call to the Moneris MPI API and passes an ACS request type.

11. Moneris MPI receives the ACS request and authenticates the information. The Moneris MPI then provides a CAVV value to the merchant

12. The merchant retrieves the CAVV value and formats a purchase or preauth request using the method that is normally used. As part of this transaction method the merchant must pass the CAVV value.

3. How do I send Transactions?

A. The TXN Request

The txn request sends the initial transaction data to the Moneris MPI to verify if the card is enrolled. Below is the code that is used to make a request to the Moneris MPI API.

```
order_id = Request.Form( "order_id" )
store_id = Request.Form( "store_id" )
api_token = Request.Form( "api_token" )
charge_total = Request.Form( "charge_total" )
pan = Request.Form( "pan" )
expiry_date = Request.Form( "expiry_date" )
md = order_id & ";" & store_id & ";" & api_token & ";" & charge_total & ";" & pan & ";" &
expiry_date & ";"

Set out = Server.CreateObject( "Moneris.Request" )
out.initMpiRequest "https://216.220.63.75:43924/mpi/servlet/MpiServlet"

accept = Request.ServerVariables( "HTTP_ACCEPT" )
useragent = Request.ServerVariables( "HTTP_USER_AGENT" )

Set purreq = Server.CreateObject( "Moneris.MPIReq" )

'Validate Order ID.
s = len( order_id )
if s <= 20 AND s > 0 then
       if s < 20 then
              order_id = order_id & "-"

              d = 20 - len( order_id )
              for i = 1 to d
                     order_id = order_id & "0"
              next
       end if

       out.setRequest  purreq.formatRequest( store_id, api_token, purreq.formatTxnRequest(
order_id, charge_total, pan, expiry_date, md,
"http://199.243.99.61/comasp/test_purchase_VBV.asp", accept, useragent ) )

       out.sendRequest
       mpiSucc = out.getMPISuccess
       mpiMsg = out.getMessage

'code continues below
```

B. The TXN Response & Creating the Popup

The txn request will return a response with several values. The "MPIsuccess" will contain "true" or "false". If the success is "false" the purchase or preauth can be sent as a crypt type 6 – attempted authentication. If success is "true" then a call to the API to create the VBV form is made.

```
       if mpiMsg = "Y" then
              Response.Write out.getMPIInlineForm     'Creates popup window on customer's
       machine.
       else
```

```
		'Not VBV Transaction, send a regular financial transaction with crypt type 7 for
	transactions with a getMessage = U or crypt type 6 for transactions with a getMessage =
	N.
		 Response.write "Popup Window getMPISuccess NOT true <br><br>"	'Debug code.
		Response.Write out.dumpXMLResponse	'Debug code.
	end if
	else
			Response.write "Order ID must be between 1 to 20 characters in length!"
	end if
	end if {
		out.print(mpiRes.getMessage()); // SEND CRYPT TYPE 6 TRANSACTION
		}
```

C. The ACS Request

After the cardholder completes the authentication the browser will return a response to the URL specified in the TermURL field that was provided in the original request. This will contain a PARes as well as a success field. Once the PARes is received it must be passed to the Moneris MPI API as a type ACS.

```
	md = Request.Form( "MD" )
	order_id = getMDComponent( md, 1 )
	store_id = getMDComponent( md, 2 )
	api_token = getMDComponent( md, 3 )
	charge_total = getMDComponent( md, 4 )
	pan = getMDComponent( md, 5 )
	expiry_date = getMDComponent( md, 6 )

	Set out = Server.CreateObject( "Moneris.Request" )
	out.initMpiRequest "https://216.220.63.75:43924/mpi/servlet/MpiServlet"

	Set purreq = Server.CreateObject( "Moneris.MPIReq" )

	out.setRequest purreq.formatRequest( store_id, api_token, purreq.formatAcsRequest(
Request.Form( "PaRes" ), md ) )

	out.sendRequest
	mpiSucc = out.getMPISuccess
```

D. The ACS Response and forming a transaction

The ACS response will contain the CAVV value. This value needs to be passed to the transaction engine along with the rest of the purchase or preauth request. Please see the documentation provided by your payment solution.

```
if mpiSucc = "true" then
		'Financial transaction with MPG.
		cavv = out.getMPICavv

		Set out = Server.CreateObject( "Moneris.Request" )
		out.initRequest store_id, api_token,
"https://216.220.63.75:43924/gateway2/servlet/MpgRequest"

		Set purreq = Server.CreateObject( "Moneris.cavvpurchase" )

		out.setRequest purreq.formatRequest( order_id, charge_total, pan, expiry_date, cavv )
		out.sendRequest

		'Display financial transaction result.
		Response.Write "Receipt ID:  " & out.getReceiptID & "<br>"
```

```
                Response.Write "Response Code:  " & out.getResponseCode & "<br>"
                Response.Write "Transaction Type:  " & out.getTransType & "<br>"
                Response.Write "Message:  " & out.getMessage & "<br>"
                Response.Write "Purchase Amount:  " & out.getTransAmount & "<br>"
                Response.Write "Card Type:  " & out.getCardType & "<br>"
                Response.Write "Reference Number:  " & out.getReferenceNum & "<br>"
                Response.Write "Transaction ID:  " & out.getTransID & "<br>"
        else
                'Authentication of cardholder by financial institution failed.  Respond to this
appropriately here.
                Response.write "ACS getMPISuccess NOT true <br><br>" 'Debug code.
                Response.Write out.dumpXMLResponse    'Debug code.
        end if
```

4. How Do I Test My Solution?

When testing your implementation of the Moneris MPI you can use the VISA PIT (production integration testing) environment to test. For testing the with the PIT we have a special store_id and api_token

Store_id : moneris
Api_token: hurgle

When testing the process is a little different in that when the popup is generated it will not contain any input boxes but rather a window of data and a submit button. When you hit submit it will load the response in the window and not in the main as it will in production.

5. How Do I Configure My Store For Production?

Once you have completed testing using the PIT you can move your store to production. You will need to change the URL from https://esqa.moneris.com:43924/mpi/servlet/MpiServlet to https://www3.moneris.com:43924/mpi/servlet/MpiServlet. As well you will need to change the store_id and api_token to the production values for your store.

6. How Do I Get Help?

If you require technical assistance while integrating your store, please contact the eSelectplus Helpdesk:
Phone: 1-888-248-3547

Email: eSelectplus@moneris.com

When sending an email be sure to include your name and phone number as well as a clear description of the problem as well as the type of API that you are using. **For security reasons, please do not send us your API Token combined with your store id, or your merchant number and device number in the same email.**

7. Appendix C. Definitions of Required Fields

Required Fields		
Variable Name	**Size/Type**	**Description**
store_id	12 / an	A value that identifies your company when your send a transaction.
api_token	20 / an	A unique key that when matched with your store_id creates a secure method of authenticating your store_id
xid	20 / an	Must be 20 alpha numeric characters. This must be unique for every transaction attempt – this can also be used as your order_id when using eSelect plus
amount	9 / decimal	Amount of the transaction. This must contain at least 3 digits including two penny values. The minimum value passed can be 0.01 and the maximum 9999999.99
pan	20 / num	Credit Card Number - no spaces or dashes. Most credit card numbers today are 16 digits in length but some 13 digits are still accepted by some issuers. This field has been intentionally expanded to 20 digits in consideration for future expansion and/or potential support of private label card ranges.
expdate	4 / num	Expiry Date - format **YYMM** no spaces or slashes. PLEASE NOTE THAT THIS IS REVERSED FROM THE DATE DISPLAYED ON THE PHYSICAL CARD WHICH IS MMYY
MD	1024 / an	This is information that you would like echoed back in the response
merchantUrl		This is the URL to which you would like the MPI response sent to.
accept		MIME types the browser accepts
userAgent		The browser details
PaRes		This is a value that is passed back to the API during the TXN and returned to the MPI when an ACS request is made.

8. Appendix D. Definitions of Response Fields

Response Fields		
Variable Name	**Size/Type**	**Description**
type	99 / an	VERes or PARes or error defines what type of response you are receiving
success	true/false	Returns whether the attempt was succesful or not
message	alpha	Will contain: a value – if it contains Y or A proceed with cavv purchase or preauth if it contains U send a crypt type 6 as a normal purchase or preauth if it contains an N or error message the transaction must be cancelled
PARes	n/a	Variable Length – data that Visa passes and needs for authentication
TermUrl	255 / an	The URL to which the PARes is returned
MD	1024 / an	Merchant defined data that will be echoed back
ACSUrl	255 / an	URL that will for the generated popup

cavv	28 /an	Visa authentication field