Himalayan Bank Payment Gateway Merchant Integration Guide

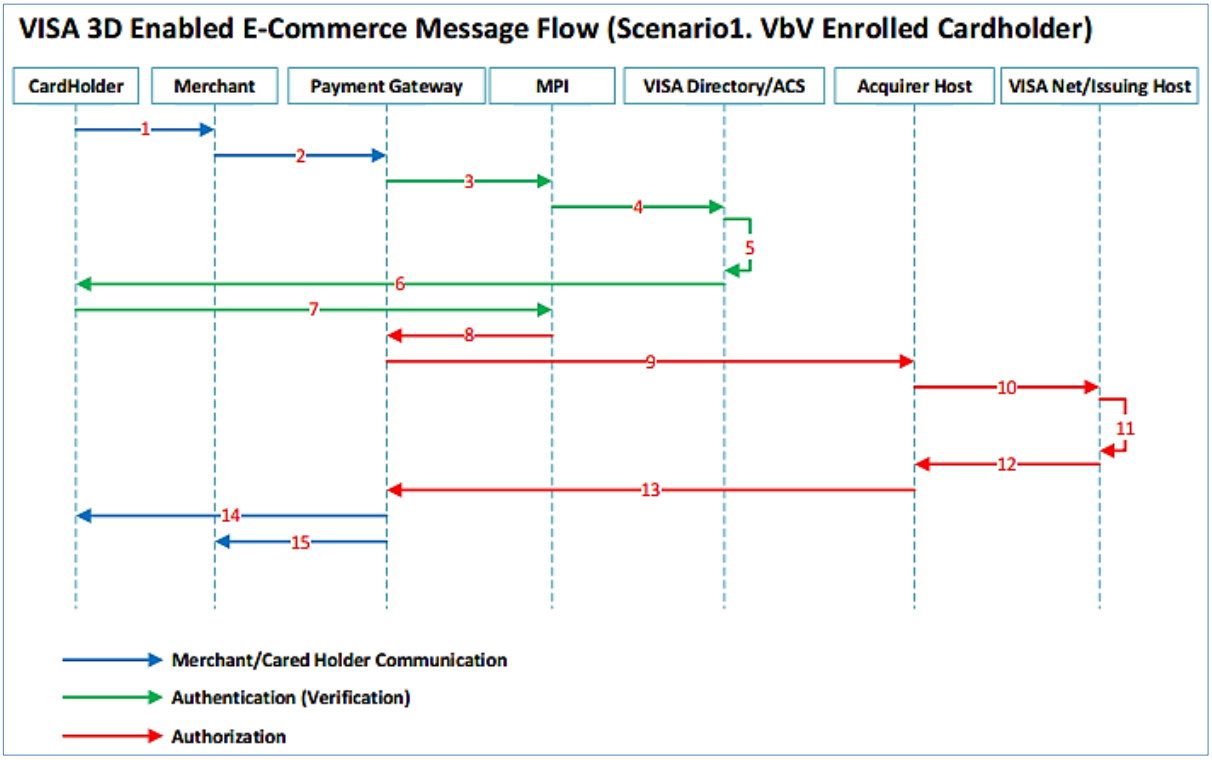
Version 1.0

Document Control		
Reference		
Author	Himalayan Bank	
Revision History		
Revision	Date	Comments/Reasons
1.0	15 September 2016	First Version

Contents

1. Message Flow (3-D Secure Environment) 4
2. Payment Request/Response 6
4. Preparing and sending payment request message using HTTP form post 9
5. Hashing 10
Appendix A - Result Code Table 13
Appendix B – Transaction Status Table 15
Appendix C – Fraud Table 16
Appendix D – Currency 16
Appendix E – Payment Warning Message 16

1. Message Flow (3-D Secure Environment)



Transaction with 3d secure enabled credit/debit card

1. Cardholder visits merchant website and completes purchase and proceed for payment.
2. During checkout process merchant website prepare payment request to HBL payment gateway and customer will see bank payment page to enter card information.
3. Once cardholder has fill all card details and confirm payment. Payment gateway proceed with 3D secure authentication using MPI (merchant plugin). Initial 3D secure message from MPI is Verify enrollment request to card scheme directory server to see if cardholder issuing bank is able to perform authentication.
4. Initial 3D secure message from MPI is Verify enrollment request to card scheme directory server to see if cardholder issuing bank is able to perform authentication.
5. Card scheme directory server forward Verify enrollment request to issuing bank in case issue is participating in 3D secure. Issuer then response Verify enrollment response back to directory what is redirected back to MPI.
6. Second step of authentication is payer authentication request. MPI post cardholder to issuing bank ACS page to complete authentication.
7. Cardholder enters one time password in bank website and ACS prepare authentication result back to MPI.
8. MPI receives authentication result from ACS and proceed with standard authorization
9. Authorization request to charge the transaction amount for the card
10. Credit card host sends out authorization message to card scheme network
11. Transaction is routed to issuer system for approval
12. Card scheme sends authorization response back to acquiring system

13. Payment gateway receives result of authorization
14. HBL-Payment Gateway display payment result screen to user
15. HBL-Payment Gateway post transaction results to Merchant with frontend result and backend response

2. Payment Request/Response

i. Field description of Payment Request

No	Variable	Data Type	Length	Mandatory	Description	Remark
1.	paymentGatewayID	Character	15	Y	paymentGatewayID	Payment Gateway ID as stated in HBL PGW Merchant Reporting.
2.	invoiceNo	Character	20	Y	Invoice No	Transaction unique invoice number. Provided by merchant.
3.	productDesc	Character	50	Y	Product Description	The product description which is provided from merchant.
4.	Amount	Numeric	12	Y	The amount of the transaction.	The amount will be padded with '0' from the left and include no decimal point. Example: 1 = 000000000100, 1.5 = 000000000150
5.	currencyCode	Numeric	3	Y	Standard ISO4217 currency codes.	Refer to Appendix D
6.	userDefined1	Character	150	N	Merchant Defined information	(Optional) HBL payment gateway will response back to merchant whatever information include in request message of this field.
7.	userDefined2	Character	150	N	Merchant Defined information	(Optional) HBL payment gateway will response back to merchant whatever information include in request message of this field.
8.	userDefined3	Character	150	N	Merchant Defined information	(Optional) HBL payment gateway will response back to merchant whatever information include in request message of this field.
9.	userDefined4	Character	150	N	Merchant Defined information	(Optional) HBL payment gateway will response back to merchant whatever information include in request message of this field.
10.	userDefined5	Character	150	N	Merchant Defined information	(Optional) HBL payment gateway will response back to merchant whatever information include in request message of this field
11.	Nonsecure	Character	1	N	Y= Payment without 3D Secure authentication N= Payment with 3D Secure authentication (default if no value present)	This parameter is optional to indicate payment without 3D secure authentication.
12.	hashValue	Character	150	N	Hash value computed by SHA-2 with secret key provided by HBL.	Refer to Hash section

ii. Field descriptions of Payment Response from payment gateway

Payment gateway send payment result back to merchant with frontend result using user browser and backend result using merchant backend URL.

It is highly recommended that merchant does not update any order/purchase status based on the frontend result. There's risk of communication breakdown from payment page to merchant. Therefore merchant should only update order/purchase status based on backend result response.

Frontend result should be used only to display landing page at merchant website.

Note. Always verify hash value of the payment gateway response to prevent data manipulation and man in the middle attacks.

No	Variable	Data Type	Length	Mandatory	Description	Remark
1.	paymentGatewayID	Character	15	Y	paymentGatewayID	Payment Gateway ID as stated in HBL PGW Merchant Reporting.
2.	respCode	Character	2	Y	The code whether the transaction is successful or not.	00 = **Approved**; 05 = Do Not Honor; etc. Refer to Appendix A for all result code.
3.	fraudCode	Character	2	Y	The code whether the fraud is passed or not.	00 = High possibility of no fraud; etc. Refer to Appendix C for all fraud code.
4.	Pan	Character	16	Y	Masked Credit Card Number	First 4 and last 4 digits of credit card number Example: 4444xxxxxxxx1111
5.	Amount	Numeric	12	Y	The amount that the customer want to convert. The amount from the request.	The amount will be padded with '0' from the left and include no decimal point. Example: 1 = 000000000100, 1.5 = 000000000150
6.	invoiceNo	Character	20	Y	Invoice No. The invoice no from the request.	Provided by Merchant. The invoice number needs to be unique to trace the transaction. Please pad '0' to the left in the case of generated invoice number length is shorter than 20
7.	tranRef	Character	28	Y	Transaction Reference for HBL.	Issued by HBL PGWSystem.
8.	approvalCode	Character	6	Y	Transaction Approval Code from Credit Card Host	
9.	Eci	Numeric	2	Y	ECI value.	
10.	dateTime	Numeric	14	Y	Process Date Time value with yyyyMMddhhmmss format	

11.	Status	Character	1	Y	Last status of transaction	AP= Approved; Refer to Appendix B
12.	userDefined1	Character	150	N	User defined info from request message	
13.	userDefined2	Character	150	N	User defined info from request message	
14.	userDefined3	Character	150	N	User defined info from request message	
15.	userDefined4	Character	150	N	User defined info from request message	
16.	userDefined5	Character	150	N	User defined info from request message	
17.	hashValue	Character	150	Y	Hash value computed by SHA-2 with secret key provided by HBL.	Refer to Hash section

4. Preparing and sending payment request message using HTTP form post

E-Commerce merchant needs to prepare the following Payment Request Form Post message. To prepare this one, merchant must identifyURL on **method="post" action=** to

Production: https://hblpgw.2c2p.com/HBLPGW/Payment/Payment/Payment

```
<Form method="post" action=https://hbl.pgw/payment>
<input type="text" id="paymentGatewayID" name="paymentGatewayID" value="234"/>
<input type="text" id="invoiceNo" name="invoiceNo" value="0000000123456789033"/>
<input type="text" id="productDesc" name="productDesc" value="Test Product"/>
<input type="text" id="amount" name="amount" value="000000010000"/>
<input type="text" id="currencyCode" name="currencyCode" value="764"/>
<input type="text" id="userDefined1" name="userDefined1" value="custom data"/>
<input type="text" id="nonSecure" name="nonSecure" value="Y"/>
<input type="text" id="hashValue" name="hashValue"
value="94E8E91C29E73B9648011FADBAE19849B520B24B"/>
</Form>
```

After sending the above HTTP Form Post request , the merchant will have the following information regarding to the transaction, at merchant's "POSTURL" page.

```
paymentGatewayID=234&respCode=00&pan=444433XXXXXX1111&amount=000000001000&i
nvoiceNo=0000000123456789033&tranRef=123&approvalCode=234567&eci=05&dateTime=20
130430111211&status=AP&fraudCode=00&userDefined1=userDefined1&userDefined2=userDefi
ned2&userDefined3=userDefined3&userDefined4=userDefined4&userDefined5=userDefined5&ha
shValue=34E8E91C29E73B9648011FADBAE19849B520B24A
```

Merchant can process the result information by using the following form request methods in various programming languages.

In ASP.Net in C#

```
string paymentGatewayID = "",respCode = "",pan = "",amount = "",invoiceNo =
"",tranRef = "",approvalCode = "", eci = "",dateTime = "",status = "";

paymentID = Request.Form["paymentGatewayID"];
respCode = Request.Form["respCode"];
pan = Request.Form["pan"];
```

In Java J2EE

```
String paymentGatewayID = "",respCode = "",pan = "",amount = "",invoiceNo =
"",tranRef = "",approvalCode = "", eci = "",dateTime = "",status = "";

paymentID = request.getParameter("paymentGatewayID");
respCode = request.getParameter("respCode");
pan = request.getParameter("pan");
```

In PHP

```
$paymentGatewayID = "", $respCode = "", $pan = "", $amount = "", $invoiceNo =
"", $tranRef = "", $approvalCode = "", $eci = "", $dateTime = "", $status = "";
$paymentID = _REQUEST["paymentGatewayID"];
$respCode = _REQUEST["respCode"];
$pan = _REQUEST["pan""];
```

5. Hashing

In order to have data integrity and to identify the correct source of the request and response, merchant needs to send hash value together in the request and HBL PGW returns the hash value in the response.

Hash value is computed using HMACSHA256 algorithm with merchant secret key (provided by HBL to merchant).

String to hash:

Type	Signature String
Payment Request	**HashValue =** merchantID + invoiceNumber + amount + currencyCode + nonSecure
Payment Response (frontend and backend)	**HashValue =** paymentGatewayID + respCode + fraudCode + Pan + Amount + invoiceNo + tranRef + approvalCode + Eci + dateTime + Status

Sample code:

Hashing function in C#

```
private string getHMAC(string signatureString, string secretKey)
{
System.Text.UTF8Encoding encoding = new System.Text.UTF8Encoding();
byte[] keyByte = encoding.GetBytes(secretKey);
HMACSHA256 hmac = new HMACSHA256(keyByte);
byte[] messageBytes = encoding.GetBytes(signatureString);
byte[] hashmessage = hmac.ComputeHash(messageBytes);
return ByteArrayToHexString(hashmessage);
}
private string ByteArrayToHexString(byte[] Bytes)
{
StringBuilder Result = new StringBuilder(); string HexAlphabet =
"0123456789ABCDEF";
foreach (byte B in Bytes)
{
Result.Append(HexAlphabet[(int)(B >> 4)]);
Result.Append(HexAlphabet[(int)(B & 0xF)]);
}
return Result.ToString();
}
```

Hashing function in PHP

```
<?php
	$signData = hash_hmac('SHA256', "signatureString",'SecretKey', false);
	$signData = strtoupper($signData);
	echo urlencode($signData);
?>
```

Hashing function in Perl

```
#!/usr/bin/perl -w
use Digest::HMAC_SHA256 qw(hmac_SHA256_hex);
print "content-type: text/html\n\n";
my $hmac_data = "signaturestring";
my $secret_key="746D7SCHAIQ0QUZ0MRJWU0PQ3AD7PJ8B";
my $output = hmac_SHA256_hex($hmac_data, $secret_key);
print $output;
```

Hashing function in Java

```
import javax.crypto.Mac;
import javax.crypto.spec.SecretKeySpec;
public class hash_hmac_SHA256 {
	public static String hmacSHA256(String strSignatureString, String key) {
	try {
		// Get an hmac_SHA256 key from the raw key bytes
		byte[] keyBytes = key.getBytes();
		SecretKeySpec signingKey = new SecretKeySpec(keyBytes,
		"HmacSHA256");
		// Get an hmac_SHA256 Mac instance and initialize with the signing
		key
		Mac mac = Mac.getInstance("HmacSHA256");
		mac.init(signingKey);
		// Compute the hmac on input data bytes
		byte[] rawHmac = mac.doFinal(strSignatureString.getBytes());
		// Convert raw bytes to Hex
		// byte[] hexBytes = Base64Coder.encode(rawHmac);
		// Covert array of Hex bytes to a String
		return byteArrayToHexString(rawHmac);
	} catch (Exception e) {
		throw new RuntimeException(e);
	}
}
static String byteArrayToHexString(byte in[]) {
	byte ch = 0x00;
	int i = 0;
	if (in == null || in.length <= 0)
	return null;
	String pseudo[] = { "0", "1", "2", "3", "4", "5", "6", "7", "8", "9",
	"A", "B", "C", "D", "E", "F" };
	StringBuffer out = new StringBuffer(in.length * 2);
	while (i < in.length) {
		ch = (byte) (in[i] & 0xF0); // Strip offhigh nibble ch = (byte)
		(ch >>> 4);
		// shift the bits down
		ch = (byte) (ch & 0x0F);
		// must do this is high order bit is on!
		out.append(pseudo[ch]); // convert thenibble to a String //
		Character
		ch = (byte) (in[i] & 0x0F); // Strip off low nibble
		out.append(pseudo[ch]); // convert the nibble to a String //
		Character
		i++;
	}
	String rslt = new String(out); return rslt;
}
```


Appendix A - Result Code Table

Result Code	Result Description
00	**Approved (transaction is successfully paid)**
01	Refer to Card Issuer
02	Refer to Issuer's Special Conditions
03	Invalid Merchant ID
04	Pick Up Card
05	Do Not Honour
06	Error
07	Pick Up Card, Special Conditions
08	Honour with ID
09	Request in Progress
10	Partial Amount Approved
11	Approved VIP
12	Invalid Transaction
13	Invalid Amount
14	Invalid Card Number
15	No Sun Issuer
16	Approved, Update Track 3
17	Customer Cancellation
18	Customer Dispute
19	Re-enter Transaction
20	Invalid Response
21	No Action Taken
22	Suspected Malfunction
23	Unacceptable Transaction Fee
24	File Update not Supported by Receiver
25	Unable to Locate Record on File
26	Duplicate File Update Record
27	File Update Field Edit Error
28	File Update File Locked Out
29	File Update not Successful
30	Format Error
31	Bank not Supported by Switch
32	Completed Partially
33	Expired Card - Pick Up
34	Suspected Fraud - Pick Up
35	Contact Acquirer - Pick Up
36	Restricted Card - Pick Up
37	Call Acquirer Security - Pick Up
38	Allowable PIN Tries Exceeded
39	No Credit Account
40	Requested Function not Supported
41	Lost Card - Pick Up

42	No Universal Amount
43	Stolen Card - Pick Up
44	No Investment Account
45	Settlement Success
46	Settlement Fail
47	Reserved
48	Cancel Fail
49	No Transaction Reference Number
50	Host Down
51	Insufficient Funds
52	No Cheque Account
53	No Savings Account
54	Expired Card
55	Incorrect PIN
56	No Card Record
57	Trans. not Permitted to Cardholder
58	Transaction not Permitted to Terminal
59	Suspected Fraud
60	Card Acceptor Contact Acquirer
61	Exceeds Withdrawal Amount Limits
62	Restricted Card
63	Security Violation
64	Original Amount Incorrect
65	Exceeds Withdrawal Frequency Limit
66	Card Acceptor Call Acquirer Security
67	Hard Capture - Pick Up Card at ATM
68	Response Received Too Late
69	Reserved
70	Settle amount cannot more than authorized amount
71	Inquiry Record Not Exist
72	Reserved
73	Reserved
74	Rejected by Fraud***
75	Allowable PIN Tries Exceeded
76	Invalid Credit Card Format
77	Invalid Expiry Date Format
78	Invalid Three Digits Format
79	Only Full Authentication Allowed***
80	User Cancellation by closing Internet Browser
81	Corporate Card Blocked***
82	Verify Request Data Failed***
83	Merchant Currency Mismatched***
84	Reserved
85	Reserved
86	Reserved

87	No Envelope Inserted
88	Unable to Dispense
89	Administration Error
90	Cut-off in Progress
91	Issuer or Switch is Inoperative
92	Financial Institution not Found
93	Trans Cannot be Completed
94	Duplicate Transmission
95	Reconcile Error
96	System Malfunction
97	Reconciliation Totals Reset
98	MAC Error
99	System Unavailable***

*** Note: Red items are the result codes which are defined by HBL payment gateway.

Appendix B – Transaction Status Table

Status	**Result Description**
AP	Approved(Paid)
SE	Settled
VO	Voided (Canceled)
DE	Declined by the issuer Host
FA	Failed
PE	Pending
EX	Expired
RE	Refunded
RS	Ready to Settle
AU	Authenticated
IN	Initiated
FP	Fraud Passed
PA	Paid (Cash)
MA	Matched (Cash)

Appendix C – Fraud Table

Status	Result Description
00	High possibility of no fraud
86	Merchant in whitelist(entry date : [[DDMMYY]])
87	PAN in whitelist(entry date : [[DDMMYY]])
88	Not Local IP Country
89	Bank Name not matched
90	Bank Country not matched
91	Exceeded over [[limit]] Txn limit of one IP using multiple PAN within 24 hours
92	Exceeded over [[limit]] PAN limit of inter non-3DS cards within 24 hours
93	Exceeded over [[limit]] PAN limit of inter 3DS cards within 24 hours
94	Exceeded over [[limit]] PAN limit of local non-3DS cards within 24 hours
95	Exceeded over [[limit]] PAN limit of local 3DS cards within 24 hours
96	BIN in black list(entry date : [[DDMMYY]])
97	IP in black list(entry date : [[DDMMYY]])
98	PAN in blacklist(entry date : [[DDMMYY]])
99	General Error : [[details]]

Appendix D – Currency

CurrencyCode	Currency Name
524	Nepalese Rupee
840	United States dollar

Appendix E – Payment Warning Message

Error Code	Description
ER001	Format Error
ER002	Duplicate Invoice Number.
ER003	HashValue Mismatched.
ER004	PaymentType Not Found
ER005	Blocked Corporate Card
ER006	Wrong Merchant Payment Type
ER007	Rejected by Fraud
ER008	Error in Authorize
ER009	Update Authen Failed
ER010	Invalid Card Number
ER011	General Error
ER012	Session Lost
ER013	-
ER014	-
ER015	-
ER016	-
ER017	-
ER018	-
ER019	Email-Ordering Merchant not found
ER020	No Merchant-ID data
ER022	Invoice-No. more than 20 characters (Credit)

=====End of Document =====