Integration Document

For eComm Merchant Adapter

Version 1.2, Mar. 2014

In-Solutions Global Pvt Ltd
601, Palm Spring, Link Road,
Malad (West)
Mumbai 400064
Maharashtra, India

Phone: +91-22-67603200
www.insolutionsglobal.com

TABLE OF CONTENTS

INTRODUCTION

eComm Merchant Adapter provides merchants a low integration and customized flow driven solution to integrate their payment enabled websites and e-commerce applications with payment gateways. It is suitable for most website hosting environments as merchants can integrate payment capabilities into their application without installing or configuring any payments software.

This guide describes how to payment enable your e-commerce application or on-line store by using the functionality of the eComm Merchant Adapter.

TRANSACTION FIELDS

This chapter covers the input fields required for basic transactions with description and sample values.

1.1 LIST OF INPUT FIELDS

The list of input fields required for each transaction is listed in table 1.

Table 1

Sr. No.	Field Name	Description
1	vpc_URL	A fully qualified URL (starting with HTTPS://). It must be included in the merchant's application code to send transaction information to the eComm Merchant Adapter. https://<YOUR_vpc_URL>/ **Note:** This URL is supplied by the Payment Provider.
2	vpc_Version	The version of the eComm Merchant Adapter API being used. The current version is 1.
3	vpc_MerchTxnRef	A unique value created by the merchant. **Usage Notes:** The Merchant Transaction Reference is used as a reference key to the Payment Server database to obtain a copy of lost/missing receipts using the QueryDR function. It can also be used to identify a duplicate transaction if it is always kept unique for each transaction attempt. It can contain similar information to the vpc_OrderInfo field, but it must be unique for each transaction attempt if it is to be used properly. Typically, the vpc_MerchTxnRef is based on an order number, invoice number, timestamp, etc., but it should also reflect the transaction attempt. For example, if a cardholder has insufficient funds on their card and they are allowed to repeat the transaction with another credit card, the value may be INV1234/1 on the first attempt, INV1234/2 on the second attempt, and INV1234/3 on the third attempt. This identifier will be displayed in the Transaction Search results in the Merchant Administration portal on the Payment Server.
4	vpc_MerchantId	The unique Merchant Id assigned to a merchant by the Payment Provider. The Merchant ID identifies the merchant account against which settlements will be made.

5	vpc_OrderInfo	The merchant's identifier used to identify the order on the Payment Server. For example, a shopping cart number, an order number, or an invoice number. This identifier will be displayed in the Transaction Search results in the Merchant Administration portal on the Payment Server. The same value may be used for both vpc_OrderInfo and vpc_MerchTxnRef provided vpc_OrderInfo is unique for each transaction attempt.
6	vpc_Amount	The amount of the transaction, expressed in the smallest currency unit. The amount must not contain any decimal points, thousands separators or currency symbols. For example, $12.50 is expressed as 1250. This value cannot be negative or zero.
7	vpc_ AccessCode	Authenticates the merchant on the Payment Gateway.
8	vpc_ReturnURL	URL supplied by the merchant in a 3-Party transaction. It is used by the Payment Server to redirect the cardholder's browser back to the merchant's web site. The Payment Server sends the encrypted Digital Receipt with this URL for decryption. It must be a fully qualified URL starting with HTTP:// or HTTPS:// and if typed into a browser with Internet access, would take the browser to that web page. It is recommended that the browser is returned to an SSL secured page. This will prevent the browser popup indicating that the cardholder is being returned to an unsecured site. If the cardholder clicks 'No' to continue, then neither the merchant nor the cardholder will obtain any receipt details.
9	vpc_SecureHash	Allows the eComm Merchant Adapter to check the integrity of the Transaction Request. The vpc_SecureHash is an SHA-256 signature of a SECURE_SECRET and the parameters in the Transaction Request or Transaction Response message. The inputs are concatenated as a single string starting with the SECURE_SECRET, then each data field in ascending alphabetical order of that field's name, with no separators and no terminating character. This string is then digested, hex-encoded and sent on its way in the vpc_Secure_Hash field. For more details see Creating an SHA-256 Signature for Transactions on page 7 and remember to always store the Secure Hash secret securely (see Store Secure Hash Secret Securely on page 7). **Note:** The secure secret is provided by the Payment Provider

1.2 DETAILS OF INPUT FIELDS

The details of input fields with sample values are provided in table 2.

Table 2

Sr. No.	Field Name	Required / Optional	Data Type	Length	Sample Data
1	vpc_URL	Req	Alphanumeric	1-255	https://geniusepay.in/GeniusepayDCC/genius/do.action
2	vpc_Version	Req	Alphanumeric	1-8	1
3	vpc_MerchTxnRef	Req	Alphanumeric	1-40	ORDER958743-1
4	vpc_MerchantId	Req	Alphanumeric	1-16	TESTDCCLTD
5	vpc_OrderInfo	Opt	Alphanumeric	0-34	ORDER958743
6	vpc_Amount	Req	Numeric	1-12	1250
7	vpc_ PassCode	Req	Alphanumeric	10	FGDX5693
8	vpc_ReturnURL	Req	Alphanumeric	1-255	https://<merchants_site>/<receipt.jsp>
9	vpc_SecureSecret	Opt	Alphanumeric	32	51DA7E1F18F490169237D3FFC9D1DE5A

1.3 DATA TRANSMISSION PROTOCOL

- Data is sent via a form POST to https://vpc_URL/
- Does not support GET data transfer. The request will be rejected

1.4 CREATING AN SHA-256 SIGNATURE FOR TRANSACTIONS

The merchant code creates the SHA-256 Secure Hash value on the Transaction Request data. The eComm Merchant Adapter creates another SHA-256 Secure Hash value and sends it back to the merchant in the Transaction Response.

The Secure Hash is a Hex encoded SHA-256 output of a concatenation of all the data parameters. The order that the data parameters are hashed in is extremely important as different transactions contain different data fields so rather than giving the explicit order for each parameter, the order that parameters are hashed in should follow the following rules:

- The Secure Hash Secret is always first.

- Then all parameters are concatenated to the secret in alphabetical order of the parameter name. More specifically, the data sort should be in ascending order of the ASCII value of each parameter's name, for example, 'Card' comes before 'card'. Where one string is an exact substring of another, the smaller string should be ordered before the longer, for example, 'Card' should come before 'CardNum'.

- Fields must not have any separators between them and must not include any null terminating characters or the like. For example, if the secret is 0F5DD14AE2E38C7EBD8814D29CF6F6F0, and the Transaction Request includes only the following parameters:

Field Name	Example Value
vpc_MerchantId	MER123
vpc_OrderInfo	Order456
vpc_Amount	2995

In ascending alphabetical order, the input to the SHA-256 Secure Hash creation routine would be:

`0F5DD14AE2E38C7EBD8814D29CF6F6F0`**`2995`**`MER123`**`Order456`**

This string is then Hex encoded and then passed through the merchant's SHA-256 Secure Hash generator in the programming language the merchant is using. This output (for example, a value of f43c85acfc4e659dcc0f654c553a199797a57cb45a8f5772770271a13fbe287b) is then included in the Transaction Request using the vpc_SecureHash field.

The Virtual Payment Client also includes the vpc_SecureHash in the Transaction Response so the merchant can check the security of the receipt data. This is performed by first stripping off the vpc_SecureHash, and then performing the same steps as creating an SHA-256 Secure Hash for the Transaction Request, but using the received Transaction Response data fields instead. The received vpc_SecureHash is then compared with the SHA-256 Secure Hash calculated from the Transaction Response data.

If both SHA-256 signatures are the same, the data has not been changed in transit. If they are different the data needs to be doubled checked.

1.5 STORAGE OF SECURE HASH SECRET SECURELY

You must keep your Secure Hash Secret stored securely. Do not store your secret within the source code of an ASP, JSP, or other website page as it is common for

web server vulnerabilities to be discovered where source code of such pages can be viewed.

You should store your Secure Hash Secret in a secured database, or in a file that is not directly accessible by your web server and has suitable system security permissions.

You should change your Secure Hash Secret regularly in accordance with your company's security policy, and any time when you believe that its security may have been compromised.