



# *Virtual Payment Client*

## Integration Guide

*Version 2.0.1*

## **Disclaimer**

We may make improvements and/or changes to the products and services described in this Manual at any time.

To the fullest extent permitted by any applicable law:

- We give no warranties of any kind whatsoever in relation to this Manual including without limitation in respect of quality, correctness, reliability, currency, accuracy or freedom from error of this Manual or the products it describes. All terms, conditions, warranties, undertakings, inducements or representations whether expressed, implied, statutory or otherwise relating in any way to this Manual are expressly excluded.
- Without limiting the generality of the previous sentence neither us nor our affiliates, employees, directors, officers or third party agents will be liable to you for any direct or indirect loss or damage (including without limitation consequential punitive or special loss or damage) however arising in respect of this Manual or any failure or omission by us, even if we are advised of the likelihood of such damages occurring.
- If we have to accept any liability our total aggregate liability to you, including any liability of our affiliates, employees, directors, officers and third party agents collectively, and regardless of whether such liability is based on breach of contract, tort, strict liability, breach of warranties, failure of essential purpose or otherwise, is limited to US500.00.
- While we have no reason to believe that the information contained in this Manual is inaccurate, we accept no responsibility for the accuracy, currency or completeness of the information in this Manual.
- We do not warrant or represent that we have checked any part of this Manual that is a copy of information we have received from a third party. We are merely passing that information on to you.

## **License Agreement**

The software described in this Manual is supplied under a license agreement and may only be used in accordance with the terms of that agreement.

## **Copyright**

MasterCard owns the intellectual property in this Manual exclusively. You acknowledge that you must not perform any act which infringes the copyright or any other intellectual property rights of MasterCard and cannot make any copies of this Manual unless in accordance with these terms and conditions.

Without our express written consent you must not:

- distribute any information contained in this Manual to the public media or quote or use such information in the public media; or
- allow access to the information in this Manual to any company, firm, partnership, association, individual, group of individuals or other legal entity other than your officers, directors and employees who require the information for purposes directly related to your business.

# Contents

<b>Preface</b>	<b>7</b>
What Is the Purpose Of This Guide .....	7
Audience .....	7
Related Documents.....	7
Where to Get Help.....	7
<b>Introduction</b>	<b>9</b>
Terminology.....	10
<b>Understanding e-Payments</b>	<b>11</b>
What are e-Payments?.....	11
The Components of an e-Payment Solution .....	12
How e-Payments Transfer Funds .....	12
About e-Payment Information Flows .....	12
The Merchant Application .....	12
The Virtual Payment Client .....	13
Payment Models.....	13
Purchase Model .....	13
Authorisation/Capture Model .....	13
Integration Models and Communication Methods.....	14
3-Party Payments Integration Model .....	15
2-Party Payments Integration Model .....	15
3-Party Payments with Card Details Integration Model .....	16
<b>Preparing for Integration</b>	<b>17</b>
Prerequisites .....	17
Support material and information .....	17
Determine your integration model.....	17
Determine the Payment Model .....	17
Determine any Advanced Functionality .....	18
Obtain an E-commerce Merchant facility .....	18
Provide your Financial Institution Merchant Number, Terminal Id/s and MCC to your Payment Provider.....	18
Look up your Access Code and Secure Hash Secret in Merchant Administration.....	19
Perform a basic test transaction using the supplied example code.....	19
Determine the input and output fields .....	19
Design and implement the integration .....	20
Test your integration .....	20
Go Live .....	20
Conduct Final Pre-Production testing .....	20
Commence Live Online Payments.....	20
Virtual Payment Client Integration Guidelines .....	21
Merchant Transaction Reference (MerchTxnRef) .....	21
Virtual Payment Client Order Information (vpc_OrderInfo).....	21
Best Practices to Ensure Successful Payments .....	22
Manually check transaction results using Merchant Administration .....	22
Automatically check transaction results.....	22

Automatically check the integrity of 3 Party transactions using Secure Hash.....	23
--	----

## **Securing your Payments** **25**

---

Protecting Cardholder Information Using SSL .....	25
How do my cardholders know if my site is using SSL? .....	25
Best Practices to Ensure Transaction Integrity .....	26
Use a unique MerchTxRef for each transaction attempt .....	26
Check that the field values in the Response match those in the Request .....	26
Check for a replay of a transaction .....	26
Check for suspect transactions .....	26
Use Good Password Security for Merchant Administration.....	27
Validate the SSL certificate of the Payment Server.....	27
Additional features for 3-Party Transactions .....	28
Detect alteration of Requests and Responses using Secure Hash.....	28
Store Secure Hash Secret Securely .....	29
Using 3-D Secure Payment Authentications .....	29
CSC Response Code .....	29
Information Flow of a Payment Authentication .....	30

## **3-Party Payments** **33**

---

Purchase Information Flow in a 3-Party Payment.....	33
What the Cardholder Sees.....	34
Integrating 3-Party Payments.....	37
Handle a Digital Order .....	37
What the Payment Server does .....	38
Handle a Digital Receipt .....	38
Handle Session Variables.....	39
Additional 3 Party Functionality.....	39
3-Party Payments where the merchant collects the cardholder's card type.....	40
3-Party Payments where the merchant collects all the cardholder's card details.....	40
3-Party Payments using Verified-by-Visa and/or MasterCard Secure Code.....	40

## **2-Party Payments** **41**

---

2-Party Payments Information Flow .....	41
What the Cardholder Sees.....	42
Advanced Merchant Administration (AMA) .....	43
AMA Capture.....	43
AMA Refund.....	44
AMA Void Capture .....	44
AMA Void Refund.....	45
AMA Void Purchase.....	45

## **Basic Transactions API Reference Fields** **47**

---

3-Party Payment Transactions.....	47
DO Fields for 3-Party Payment Requests.....	47
Sending a DO for 3-Party Payments.....	49
DR Fields for 3-Party Payments .....	49
2-Party Payment Transactions.....	57
Digital Order Fields for 2-Party Payments .....	57
2 Party Style Pre-Authenticated Payment Transactions.....	61
Sending a DO for 2-Party Payments.....	61
DR Fields for 2 Party Payments.....	62
Additional Functionality API Reference Fields.....	67

<b>Troubleshooting and FAQs</b>	<b>83</b>
<hr/>	
Troubleshooting.....	83
What happens if a Transaction Response fails to come back?.....	83
What do we do if a Session Timeout occurs?.....	85
What does a Payment Authentication Status of "A" mean? .....	85
Does the Cardholder's Internet browser need to support cookies?.....	85
How do I know if a transaction has been approved? .....	85
What is an outage? .....	85
Frequently Asked Questions .....	86
Is a Shopping Cart required?.....	86
What is Merchant Administration? .....	86
How much will it cost to keep the payment site running? .....	86
Does the Payment Server handle large peaks in transaction volumes? .....	86
How long will an authorisation be valid on a cardholder account? .....	86
What is the RRN and how do I use it?.....	87
What is the difference between RRN, MerchTxnRef, OrderInfo, Authorizeld and TransactionNo ?.....	87
 <b>References - Virtual Payment Client</b>	 <b>89</b>
<hr/>	
Returned Response Codes .....	90
Card Type Code .....	91
Verified by Visa™ (Visa 3-Domain Secure) and MasterCard SecureCode™ (MasterCard 3- Domain Secure) Status Codes.....	92
Error Codes and their descriptions for the most commonly encountered errors .....	93
The complete list of Error Codes and their descriptions are:.....	94
 <b>Appendix – Test Environment</b>	 <b>103</b>
<hr/>	
Test Cards.....	103
Issuer Response Code Mapping.....	104



---

## CHAPTER 1

# Preface

---

## What Is the Purpose Of This Guide

This Merchant Developers Guide describes the Virtual Payment Client (VPC) API (Application Programming Interface) which allows you to payment enable your e-commerce application or on-line store. It seeks to guide you on how to use the functionality of the Virtual Payment Client API. The document describes Version 1.0 of the Virtual Payment Client API.

In addition, the guide outlines the business logic around payment processing on the MIGS Payment Server and how to use the VPC to perform payment processing and, if required, integrated administration functions.

---

## Audience

This guide is for developers who need to integrate a payments' solution into merchant applications.

---

## Related Documents

To complete the merchant offering, transactions processed through MIGS via the Virtual Payment Client Guide can be administered via the MIGS Merchant Administration portal. To understand what this portal offers, readers should reference the MIGS Merchant Administration Guide.

---

## Where to Get Help

If you need assistance with Virtual Payment Client Integration, please contact your support organization's help desk, the details of which you will be given once you sign up to the MIGS service via your bank.



## CHAPTER 2

# Introduction

The Virtual Payment Client enables merchants to payment enable websites, e-commerce or other applications by providing a low integration effort solution. 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 Virtual Payment Client.

The document describes Version 1.0 of the Virtual Payment Client. It details the base and supplementary fields for the different types of transactions, and includes additional material such as valid codes, error codes and security guidelines.

## Terminology

Term	Description
<b>Access Code</b>	The access code is an identifier that is used to authenticate you as the merchant while you are using the Virtual Payment Client. The access code is generated and allocated to you by the Payment Server.
<b>Acquirer Bank</b>	Where your business account is maintained and settlement payments are deposited. This is normally the same bank with which you maintain your merchant facility for your online credit card payments.
<b>Bank</b>	The bank with which you have a merchant facility that allows you to accept online credit card payments.
<b>Capture</b>	A capture is a transaction that uses the information from an authorization transaction to initiate a transfer of funds from the cardholder's account to the merchant's account.
<b>Financial Institution (FI)</b>	See Bank.
<b>Issuing Bank</b>	The financial institution that issues credit cards to customers.
<b>Merchant Administration</b>	Merchant Administration allows you to monitor and manage your electronic transactions through a series of easy to use, secure web pages.
<b>Payment Provider</b>	The Payment Provider acts as a gateway between your application or website and the financial institution. It uses the Payment Server to take payment details (Transaction Request) from your cardholder and checks the details with the cardholder's bank. It then sends the Transaction Response back to your application. Approval or rejection of the transaction is completed within seconds, so your application can determine whether or not to proceed with the cardholder's order. Your Payment Provider may be your acquirer bank or a third party technology services provider.
<b>Payment Server</b>	The Payment Server facilitates the processing of secure payments in real-time over the Internet between your application/website and the Payment Provider. All communications between the cardholder, your application, the Payment Server and the Payment Provider is encrypted, making the whole procedure not only simple and quick, but also secure.
<b>Purchase</b>	Purchase is a single transaction that immediately debits the funds from a cardholder's credit card account.
<b>RRN</b>	The RRN (Reference Retrieval Number) is a unique number generated by the payment provider for a specific merchant ID. It is used to retrieve original transaction data and it is useful when your application does not provide a receipt number.
<b>Transaction Request</b>	This is also called the Digital Order(DO) and is a request from the Virtual Payment Client to the Payment Server to provide transaction information.
<b>Transaction Response</b>	This is also called the Digital Receipt(DR) and is a response from the Payment Server to the Virtual Payment Client to indicate the outcome of the transaction.

## CHAPTER 3

# Understanding e-Payments

This section is an overview of electronic payments or e-Payments.

## In This Chapter

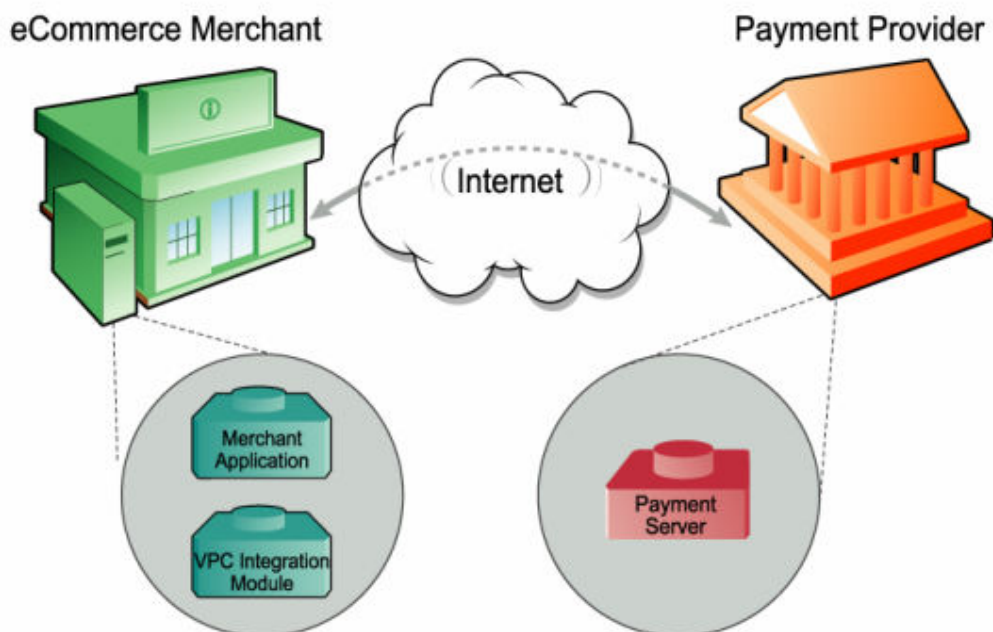
What are e-Payments? .....	11
About e-Payment Information Flows .....	12
Payment Models .....	13
Integration Models and Communication Methods .....	14

---

## What are e-Payments?

e-Payments are secure real time payments that transfer funds (via the Internet) between a consumer and the merchant's financial institutions. e-Payments require secure communication between all components of the e-Payment process.

e-Payments are represented in the following diagram:



## The Components of an e-Payment Solution

An end-to-end e-Payment solution is made up of the following components:

- **The Merchant application** is a business application/website on the merchant's system that uses Virtual Payment Client to process payments.
- **The Integration module** is a communication bridge between the merchant application and Virtual Payment Client.
- **Virtual Payment Client** provides secure communication between the merchant application and the Payment Server. Virtual Payment Client can be integrated with a number of systems including merchant applications, Interactive Voice Response (IVR) systems, and integrated ERPs
- **Payment Server** processes merchant Transaction Requests.
- **The Payment Provider** enables the merchant to accept payments online.

## How e-Payments Transfer Funds

e-Payments transfer funds via the following steps:

- 1 The cardholder purchases goods/services from the merchant (for example, in person, via the Internet, over the phone).
- 2 The merchant application sends a Virtual Payment Client Transaction Request (via the Payment Server) to the merchant's Payment Provider.
- 3 The merchant's Payment Provider directs the request to the cardholder's bank.
- 4 The cardholder's bank debits the cardholder's account and transfers the funds to the merchant's account at the merchant's Payment Provider.

---

## About e-Payment Information Flows

This section describes how information is transferred between the merchant application and the Payment Server.

### The Merchant Application

To process a payment, the merchant application must send the required information to the Payment Server. The merchant application must create a message in a specified format to send this information via the Virtual Payment Client, which is part of the Payment Server using two messages:

- **Transaction Request** is sent to the Virtual Payment Client in the Payment Server to provide transaction information.
- **Transaction Response** is returned from the Payment Server via the Virtual Payment Client to indicate the outcome of the transaction (that is, successful or otherwise).
- A **Transaction** is the combination of a Transaction Request and a Transaction Response. For each customer order, merchants may issue several transactions.

## The Virtual Payment Client

- Receives the Transaction Request from the merchant application; and
- Sends the information to the Payment Server
- The Virtual Payment Client receives the result from the Payment Server, creates a response in the appropriate format and forwards it to the Merchant Application.

---

## Payment Models

Virtual Payment Client supports the most commonly used payment models in the e-Payments process. These are:

- Purchase Model
- Authorisation/Capture Model

Payment Integration models are described in *Preparing for Integration* (see "Preparing for Integration" on page 17).

### Purchase Model

Purchase is the most common type of payment model used by merchants to accept payments. A single transaction is used to authorise the payment and initiate the debiting of funds from a cardholder's credit card account.

This is typically used when the goods will be delivered immediately following a successful transaction.

### Authorisation/Capture Model

The authorisation/capture payment type is a two step process. The merchant uses an Authorisation transaction to reserve the funds.

#### Authorisation in the Auth/Capture Model

The Authorisation (Auth) transaction verifies that the card details are correct and may/may not also reserve the funds, depending on the merchant's Payment Provider. To find out what models are available to you, we recommend that you discuss it with your Payment Provider.

The authorisation is used to ensure that the cardholder has sufficient funds available against their line of credit. The full amount of the order is sent to the card Issuing Bank to verify the details against the cardholder's card account. The authorisation does not debit funds from the cardholders account, but reserves the total amount, ready for the capture transaction to debit the card and transfer the funds to your account.

The cardholder's credit limit is reduced by the authorised amount. If they make another transaction, this current authorisation transaction is taken into account and comes off the cardholder's available funds as though the transaction had already taken place. This authorisation reserves the funds for a predetermined period of time, (such as 5-8 days), as determined by the card scheme and the cardholder's card issuing rules.

The API does not have a method to void an Authorisation transaction so it must fade out at the end of the appropriate period. Authorisation transactions do not appear in the cardholder's account records, only the capture transactions appear.

The Authorisation transaction uses the same API as the standard payment transaction used in the Purchase model where a Capture transaction is not required. The only difference is how the merchant profile is configured with the Payment Provider.

## Capture in the Auth/Capture Model

The capture transaction refers back to the initial authorisation transaction, and transfers the funds from a cardholder's card into the merchant's account.

The merchant can perform any number of capture transactions on the original Authorisation transaction, however the total of all the amounts from all the captures cannot exceed the original authorised amount. For example, the merchant may not have the full ordered amount of goods in stock. Hence they ship what they do have and capture the funds from the cardholder accordingly. Later when the remaining goods are shipped the merchant performs another capture transaction that refers back to the same initial authorisation transaction. This causes the remaining funds to be transferred from the cardholder's account to the merchant's account. The capture transactions will be successful, provided:

- The total amount for the all captures do not exceed the original Authorisation amount, and
- The card issuing institution has not expired the original Authorisation transaction.

---

## Integration Models and Communication Methods

There are two ways that you can communicate with the Payment Server to process transactions, the Redirect method and the Direct method. The method you choose is directly related to the Integration Model (3-Party Payments or 2-Party Payments) that you use. You may use both methods concurrently if you desire, e.g. you may have a Web Store that uses 3-Party payments, and at the same time a Call Centre taking phone orders using 2-Party payments. Both applications could be using the Payment Client at exactly the same time.

For web applications, you must have an SSL certificate from a trusted certificate issuer for encrypting the data between the cardholder's browser and the merchant's web site. For other applications the method of security will vary. However, you can securely gather card details and pass them into a transaction without storing them in a database.

In terms of data storage, both the 2-Party and 3-Party with card details integration models place certain obligations that you would be required to comply with if you store these card details. Your financial institution best defines the security requirements for gathering and storing card details.

## 3-Party Payments Integration Model

3-Party Payments allow the Payment Server to manage the payment pages and collect the cardholder's card details on your behalf. The Payment Server's payment pages could be Bank or Payment Provider branded to help assure the cardholder of a secure transaction. The advantage of 3-Party payments is that the complexity of securely collecting and processing card details is handled by the Payment Server, allowing you to focus on your application's part of the payment process.

However, 3-Party Payments do also allow you to collect card details on your web site and pass them through with the other transactional details. If this is done the Payment Server does not display any 3rd party branded pages, keeping the branding consistent throughout the whole transaction, except the 3-D Secure pages if the merchant and the cardholder are both enrolled in this antifraud initiative. To do this you would have to comply with the same obligations associated with 2-Party payments.

The 3-Party Redirect method only works for web applications where a web browser is involved. This method is also required to implement 3-D Secure antifraud initiatives of Verified by Visa™ (Visa 3-Domain Secure) and MasterCard SecureCode™ (MasterCard 3-Domain Secure). The redirect method works with most network configurations and you do not need to take into account proxy servers as the information is communicated to and from the Payment Server via the cardholder's Internet browser.

The cardholder's Internet browser is redirected to take the Transaction Request to the Payment Server to process the transaction. After processing the transaction, the cardholder's Internet browser is returned to a web page that you nominate in the transaction together with a Transaction Response. The Transaction Response processing of the receipt information finalises the transaction.

The 3 parties involved in a 3-Party transaction are the merchant, the payment provider and the cardholder. The cardholder's browser provides the redirect method to communicate the information between the merchant and the payment provider. This is an a-synchronous connection and the cardholder leaves your web site to go to the Payment Server, which means the transaction is broken or disrupted into 2 distinct sessions, the creation of the Transaction Request and the processing of the Transaction Response.

Because of this, you may be required to capture session variables and include them in the Transaction Request so they can be passed back appended to the Transaction Response for restoring the original web session.

## 2-Party Payments Integration Model

Merchants who want full control over the transaction and want to manage their own payment pages use the 2-Party/Direct Payments integration model. Implementing 2-Party payments requires you to securely collect the cardholder's card details and then use the Virtual Payment Client to send the Transaction Requests directly to the Payment Server. This is also called the merchant-managed, or direct model. This model means that you are responsible for securing the cardholders card number and details.

The 2-Party or direct model does not allow you to implement the 3-D Secure anti-fraud initiatives of Verified by Visa™ (Visa 3-Domain Secure) and MasterCard SecureCode™ (MasterCard 3-Domain Secure).

The 2 parties involved in a 2-Party transaction are the merchant and the payment provider. The merchant communicates directly through the Virtual Payment Client to the Payment Server and back again. This is a synchronous connection and the cardholder does not leave your site, which means the session is not broken or disrupted.

The Direct method is also used for advanced Payment Server operations such as captures, refunds voids and queries. Your application communicates to the Payment Server via the Virtual Payment Client, so you need to take into account working with proxy servers.

The methods used to work with these proxy servers will vary slightly depending on the programming language used by your application.

## **3-Party Payments with Card Details Integration Model**

If you require the 2-Party Payments Integration Model and the 3-D Secure anti-fraud scheme, you will have to implement the 3-Party Payment /w Card Details Integration Model.

Basically in this model you are initializing a normal 3-Party Payment transaction but at the same time you, as the merchant, also collect and send the card details to the Payment Server. This results in the card selection and card details collection procedures to be bypassed, skipping directly to the 3-D Secure Authentication stage.

For more details please refer to "Additional 3 Party Functionality" at page 39.

## **The Direct Communication Method**

The Direct method is also used for advanced Payment Server operations such as 2-Party Purchases and Auths, Captures, Refunds Voids and Queries. Your application communicates directly to the Payment Server via the Virtual Payment Client, so you need to take into account working with proxy servers. The methods used to work with these will vary slightly depending on the programming language used by your application, however the Virtual Payment Client has the ability to work with HTTP, Socks4 and Socks5 proxy servers.

---

## CHAPTER 4

# Preparing for Integration

Before you start integrating, you must determine if your Payment Provider supports the functions that you require. This will determine the transaction types you can or cannot integrate.

---

## Prerequisites

This section lists the requirements and basic steps you need to take to build a successful integration.

## Support material and information

You must have the following:

- Virtual Payment Client Reference Guide
- Example Code for your site (written in ASP, JSP, and PHP)

## Determine your integration model

You must choose either:

- 3-Party Payments Integration Model, or
- 2-Party Payments Integration Model, or
- 3-Party Payments /w Card Details Integration Model, or
- Combination of the above

For more information, please refer to *Integration Models* (see "Integration Models and Communication Methods" on page 14).

## Determine the Payment Model

- **Purchase** - requires a single transaction to transfer funds from the cardholder's account to your account.
- **Authorisation/Capture** - requires two transactions, the Authorisation, followed separately by a Capture. For more information, see *Authorisation/Capture* (see "Authorisation/Capture Model" on page 13).

## Determine any Advanced Functionality

The available advanced functionality includes:

- Verified by Visa™ (Visa 3-Domain Secure) and MasterCard SecureCode™ (MasterCard 3-Domain Secure). See **Securing your Payments** (see "Securing your Payments" on page 25).
- **Capture** (see "Capture" on page 67).
- **Refund** (see "Refund" on page 70).
- **Voids** (see "Void Capture" on page 72).
- **QueryDR** (see "QueryDR Transaction" on page 81).

## Obtain an E-commerce Merchant facility

After your E-Commerce merchant facility has been approved, your Financial Institution (FI/Bank) will provide the following information to you:

- **Bank Merchant Number (CAIC)**  
Without this information you cannot perform any transactions. It **ensures** your settlement funds from successful payments are deposited to your correct account.  
**IMPORTANT:** The CAIC is **not** the same as the Merchant ID, although they **can** be the same.
- **Terminal Id/s**  
The identifier of the computer terminal/s that you will be using to process payments.
- **Merchant Category Code (MCC)**  
A four-digit code allocated to you by the Payment Provider based on your business type.

## Provide your Financial Institution Merchant Number, Terminal Id/s and MCC to your Payment Provider

(NOTE: Your Financial Institution can also be your Payment Provider, this is a common case.)

This information is needed to establish your merchant profile with your Payment Provider. Your merchant profile holds your configuration data including Financial Institution account details and your access credentials to the payments service.

Your Payment Provider will then issue you with a unique Payment Server Merchant Id identifying you to the Payment Server and also provide you with a User Name and Password for accessing Merchant Administration to manage your transactions.

## Look up your Access Code and Secure Hash Secret in Merchant Administration

You need your Virtual Payment Client Access Code and Secure Hash Secret before starting your integration:

- **Access Code**  
The access code uniquely authenticates a merchant and their Merchant Id on the Payment Server.
- **Secure Hash Secret**  
If you are using 3-Party Payments, the Secure Hash Secret is a key used as the initial piece of encryption data to create an MD5 Secure Hash to ensure transaction data is not tampered with while in transit to the Virtual Payment Client

Your access code and secure hash secret can be found in Merchant Administration in the Setup menu option on the Configuration Details page. Please refer to your Merchant Administration User Guide for details on how to locate your Access Code and Secure Hash Secret

## Perform a basic test transaction using the supplied example code

Successful completion of a transaction using the standard Dialect example code before you implement the integration with your application:

- Validates that your system is set-up correctly and
- Ensures basic functionality is available.

The standard example code covers common web server scripting languages. You must select the appropriate example for your specific web environment.

**Note:** The standard example code contains examples of how to integrate your application and may not fully correspond with the feature set that you have chosen to implement.

## Determine the input and output fields

Determine how you are going to get the Transaction Request input fields and where to store the Transaction Response output fields in your application.

You need to consider:

- **Session Variables** - When using 3-Party payment (with or without card details) some applications may require session variables to be collected and sent to the Payment Server in the Transaction Request. The session variables are returned in the Transaction Response allowing your application to continue with the order process using the same application session. For more information on session variables see, **Session Variables** (see "Handle Session Variables" on page 39).

Session variables are not required when using the Direct or 2-Party communication method as the session is not broken while performing a transaction.

- **Merchant Transaction Reference (vpc\_MerchTxnRef)** - You need to determine how you are going to produce a unique value for a transaction using the **vpc\_MerchTxnRef** field. For more, see Merchant Transaction Reference.

## Design and implement the integration

You are now ready to payment enable your application. This step requires a web developer familiar with both your application and the web programming language used in your web environment.

This guide provides the information and best practice guidelines to assist you with this task. You should also refer to the example code and Virtual Payment Client Reference Guide for further assistance.

## Test your integration

You need to test your integration by performing test transactions. The Payment Server has a test acquirer facility to test all the different response codes that you are likely to encounter in a live environment.

Performing test transactions allows you to test your integration, so that you won't encounter problems when processing real transactions. For more information, please refer to the Test Card set up document supplied to you by your Payment Provider.

## Go Live

Once you are satisfied that your integration works correctly, please advise your Payment Provider that your testing has been successfully completed.

Your Payment Provider will validate your testing results and then provide you with your production profile and instructions on how to change your website from test mode to live production mode, allowing you to process live transactions with your Financial Institution (bank).

## Conduct Final Pre-Production testing

It is recommended that you follow standard IT practices and complete final pre-production testing with live credit cards to validate that end-to-end functionality works correctly, including successful settlement of funds from your financial institution.

Remember you can always refund these test transactions, which is also testing the refund capability is working correctly.

## Commence Live Online Payments

You should now be ready to launch your payment enabled application and start processing online payments from your cardholders.

---

# Virtual Payment Client Integration Guidelines

This section describes certain key issues that you must take into account while writing your integration code.

## Merchant Transaction Reference (MerchTxnRef)

The *vpc\_MerchTxnRef* field is a unique identifier that the merchant assigns to each transaction. This unique value is used by the merchant to query the Payment Server database to retrieve a copy of a lost/missing transaction receipt using a 2-Party QueryDR function. This is the main purpose of *vpc\_MerchTxnRef*.

If you keep *vpc\_MerchTxnRef* unique, the other function of it is to retrieve the value from the Transaction Response and compare it with the transaction value in your database. You should check each transaction response to ensure that your unique Merchant Transaction Reference Id (*vpc\_MerchTxnRef*) in the Transaction Response matches that order, and that it does not correspond with any previous order that has already been processed.

You can use a value like an order number or an invoice number as the foundation for the *vpc\_MerchTxnRef*. However, if you want to allow cardholders to repeat a transaction that was declined and you want to keep the same order number (or invoice number), you must modify the *vpc\_MerchTxnRef* for each subsequent attempt, by appending extra characters for each attempt. For example *vpc\_MerchTxnRef* = '1234/1' on first attempt, '1234/2' on second attempt, and '1234/3' on third attempt, etc.

Under a fault condition, such as if the Digital Receipt does not arrive back at the merchant's site due to a communication error, you may need to check if the transaction was carried out successfully. A unique *vpc\_MerchTxnRef* makes cross-referencing the transactional data easier.

This is achieved by performing a QueryDR command that will search the Payment Server's database for the transaction, based on the *vpc\_MerchTxnRef*. If you have not given each transaction attempt a unique *vpc\_MerchTxnRef* number, then there will be multiple results and the QueryDR command may not return the correct transaction attempt you are looking for as it only returns the most recent transaction information.

## Virtual Payment Client Order Information (vpc\_OrderInfo)

**vpc\_OrderInfo** is an identifier provided by you to identify the transaction on the Payment Server database. This value will be displayed in the Merchant Administration portal when manually searching transactions. It can be an order number, an invoice number, or a shopping cart number. The **vpc\_OrderInfo** field can remain static for each transaction but you should have a unique **vpc\_MerchTxnRef** for each transaction as outlined above for use with the QueryDR function.

The **vpc\_OrderInfo** field is used to send a merchant specified reference, for example, Merchant's Invoice No = **vpc\_OrderInfo** and can be used to search for another in Merchant Administration.

---

## Best Practices to Ensure Successful Payments

It is recommended that you consult with security experts with experience in your web environment to ensure that your security implementation is suitable for your needs.

An issue that merchants have to deal with when implementing payments solutions is "How to be sure that you get paid for the goods you ship?"

To ensure that you will be paid means that you need to ensure the response integrity and identification/authentication of the Payment Server during the payment process.

To ensure that you will be paid you can:

- Where possible, implement the 3-Domain Secure services of Verified by Visa™ (Visa 3-Domain Secure) and MasterCard SecureCode™ (MasterCard 3-Domain Secure). See **Securing your Payments** (see "Securing your Payments" on page 25).
- Manually check all transaction results at the Payment Server by logging into Merchant Administration before fulfilling each order.
- Automatically check transaction results at the Payment Server before fulfilling each order by using the **Query DR** (see "QueryDR Transaction" on page 81) functionality (if available).
- Automatically check and verify the integrity of each message when the payment is performed by using the Secure Hash functionality.

### Manually check transaction results using Merchant Administration

This process suits merchants with very low volume sales. It requires you to log in to Merchant Administration and run a report to view the Payment Server OrderIDs (Shopping Transactions) and then match them against the orders logged on your website. If they match, you can ship the product, and follow up on, or discard orders where the payment failed, or the payment does not exist.

The risk is the possible human error of matching OrderIDs with the cardholder's orders manually. Also as volumes grow, the risk may become significant as would the time and cost involved completing the task.

### Automatically check transaction results

If you have implemented 3-Party Payments and a Transaction Response is not received, this method involves automatically attempting to retrieve the Transaction Response directly from the Payment Server after each transaction where the Transaction Response failed to come back. This is because in a 3-Party transaction the communication path via the customer's browser, which is redirected away from the merchant's web site. If the cardholder cancels the transaction, no Transaction Response will be returned, which is not as reliable as a direct 2-Party connection to the Payment Server. Also, the transaction may have been successful, but you may not receive a receipt as the communication link failed at the point of redirecting the cardholder's browser back to the merchant's site.

To overcome these problems QueryDR is available, but it requires some level of programming ability, and system configuration as proxies and firewalls need to be modified to allow outbound connections.

## Automatically check the integrity of 3 Party transactions using Secure Hash

The Secure Hash is used to detect the cardholder modifying a Transaction Request or Transaction Response when passing it through their cardholder's browser. Using the Secure Hash ensures a high level of trust in the transaction result.

The benefit of using Secure Hash is that the integrity of each response can be checked without having to create a new SSL connection to the Payment Server for each transaction.

The Secure Hash Secret must be kept secret to provide security and should be changed periodically for this method to be effective

The Secure Hash method is **only applicable when using the 3-Party** Payments integration model.



---

## CHAPTER 5

# Securing your Payments

This section describes the security features available for the Virtual Payment Client. It is recommended that you understand this section before you start integrating your application with the Virtual Payment Client.

---

## Protecting Cardholder Information Using SSL

All websites collecting sensitive or confidential information need to protect the data passed between the cardholder's Internet browser, the application and the Payment Server.

SSL is a security technology that is used to secure web server to Internet browser transactions. This includes the securing of any information (such as a cardholder's credit card number) passed by an Internet browser to a web server (such as your web 'Shop & Buy' application). SSL protects data submitted over the Internet from being intercepted and viewed by unintended recipients.

The Payment Server is responsible for securing the cardholder details when you implement the 3-Party Payments Integration Model. It uses SSL, which encrypts sensitive financial data to provide a secure transmission between a cardholder and the Payment Server.

When implementing the 2-Party or 3-Party Payments Integration models you must ensure your application presents a secure form using SSL. You should also consider using a secure form in your application when collecting confidential information such as cardholder addresses.

**Note:** SSL should also be used for 3-party payments. This avoids the possible browser alert message indicating that the cardholder is being redirected to an unsecured site. This can happen when the cardholder's browser is being redirected back to the merchant's web site with the encrypted Transaction Response for decryption.

If the cardholder clicks on 'No' within the pop-up message, then neither the cardholder nor the merchant will receive any receipt details.

## How do my cardholders know if my site is using SSL?

When a cardholder connects to your application using SSL they will see that the http:// changes to https:// and also a small gold padlock will appear in their Internet browser, for example:



Whenever an Internet browser connects to a web server (website) over https:// - this signifies that the communication with the Payment Server will be encrypted and secure.

You can alert your customers to this fact so they know what to look for when transacting on your web site.

## Best Practices to Ensure Transaction Integrity

The following Best Practices are guidelines only. It is recommended that you consult with security experts with experience in your web environment to ensure that your security is appropriate for your needs.

### Use a unique MerchTxRef for each transaction attempt

Each transaction attempt should be assigned a unique transaction reference Id. Most applications and web programming environments will generate a unique session for each cardholder, which can be used as the unique merchant transaction reference Id. You can alternatively create a unique reference id by combining a order/invoice number with a payments attempt counter. You may also consider appending a timestamp to the transaction reference Id to help ensure that each one is unique.

Before sending a transaction to the Payment Server, you should store this unique transaction reference Id with the order details in your database. The merchant transaction reference id is returned in the Transaction Response.

The unique transaction reference Id is required for you to reliably use the QueryDR function to retrieve the transaction details you may be searching for. For example, if a transaction is reported as lost or missing, you can use QueryDR to locate it.

### Check that the field values in the Response match those in the Request

You should ensure that important fields such as the amount and the merchant transaction reference ID in the Transaction Response match up with the values input to the original Transaction Request.

### Check for a replay of a transaction

You should check each Transaction Response to ensure that your unique Merchant Transaction Reference Id matches that order, and that it does not correspond with any previous order that has already been processed.

### Check for suspect transactions

Common things to look out for are:

- Use of free/anonymous E-mail by the cardholder
- Different Ship To and Bill To addresses
- Foreign orders or shipments from countries with reputations for high fraud activities
- High-priced orders
- Multiples of the same item.

**Note:** It is recommended that you do not store any credit card information in your web site database. If you must store credit card numbers, they should be securely hardware encrypted, or you should store them obfuscated (for example, 498765XXXXXXXX769)

## Use Good Password Security for Merchant Administration

You should choose a password that is difficult to guess and you should change your password regularly. A good password should be at least 8 characters and should contain a mix of capitals, numbers and special characters.

A useful technique to create a password is to choose a phrase such as a line from a song, or a nursery rhyme and take the first letter of each word. For example, you could create the password `Tt1*H1wwya` from the phrase, Twinkle, twinkle little star, How I wonder what you are. In the example, an asterisk character has been used for the word 'star' and the digit 1 for the capital letter 'I'.

## Validate the SSL certificate of the Payment Server

You should validate the SSL certificate of the Payment Server whenever connecting to the Payment Server. The Payment Server SSL certificate is issued by an industry standard Certificate Authority such as Verisign or Thawte whose root certificate should already be available in your web environment.

**Note:** Please consult a web developer if you are not familiar with validating SSL certificates or exporting certificates from websites.

---

## Additional features for 3-Party Transactions

The following features apply to 3-Party transactions only.

### Detect alteration of Requests and Responses using Secure Hash

The Virtual Payment Client uses MD5 Signature Hashes, which play a role in transaction security as they are used to detect whether the Transaction Request and Transaction Response has been tampered with. The Secure Hash Secret is generated by the Payment Server and assigned to you. It is a unique value for each merchant and made up of alphanumeric characters. Only you and the Payment Server know what the Secure Hash Secret value is.

Your Secure Hash Secret is used along with the Transaction Request details uses an MD5 algorithm to generate a Secure Hash, which is appended to the Transaction Request. Because the Payment Server is the only other entity apart from your application that knows your Secure Hash Secret, it is the only other entity that can recreate the same Secure Hash:

- If the Secure Hash recreated by the Payment Server is equal to the Secure Hash sent in the Transaction Request, it means that the Transaction Request has not been tampered with. The Payment Server will continue to process the payment.
- If the Secure Hash recreated by the Payment Server does not equal the Secure Hash sent in the Transaction Request it can be assumed the data has changed in transit. The Payment Server will not process the payment and return the cardholder to your site with an error message in the Transaction Response.

After processing the transaction, the Payment Server uses your Secure Hash Secret and the Transaction Response details to generate a Secure Hash which is sent to your application. Your application uses the Secure Hash Secret and the Transaction Response details received from the Payment Server to also generate a Secure Hash. If your generated Secure Hash matches the Secure Hash sent by the Payment Server the Transaction Response has not been tampered with.

If your generated Secure Hash does not match the Secure Hash sent by the Payment Server the Transaction Response has been tampered with and should be checked against the data stored in the Payment Server. This can be done by using an automatically QueryDR (if available) or manually using Merchant Administration.

The Secure Hash Secret is never sent from the application to the Payment Server or from the Payment Server to the application. It is held securely at both sites and is only used as a seed in the generation of the Secure Hash.

For more information on Secure Hash Secret, see **Store Secure Hash Secret securely** (see "Store Secure Hash Secret Securely" on page 29)

---

## Store 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 or 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 at least once a year, or any time when you believe that it's security may have been compromised.

You can change your Secure Hash secret in Merchant Administration in the Setup menu option on the Configuration Details page. For more information, please refer to your Merchant Administration User Guide.

---

## Using 3-D Secure Payment Authentications

3-D Secure Payment Authentication contains Verified by Visa™ (Visa 3-Domain Secure) and MasterCard SecureCode™ (MasterCard 3-Domain Secure), which are designed to minimise credit card fraud, by attempting to authenticate cardholders when performing transactions over the Internet. Authentication ensures that a legitimate owner is using the card as the Payment Server redirects the cardholder to their card issuing institution where they must enter a password that they had previously registered with their card issuer.

To use Verified by Visa™ (Visa 3-Domain Secure) and MasterCard SecureCode™ (MasterCard 3-Domain Secure), you need to request a 3-D Secure enabled merchant profile from your Payment Provider and implement the 3-Party Payment Integration Model.

**Note:** Payment authentication is only supported for web transactions using 3-Party Payments through a browser. This is because the cardholder's web browser must be redirected to their card issuing bank.

For the information flow of a 3-Party Authentication & Payment transaction please see Authentication Information Flow.

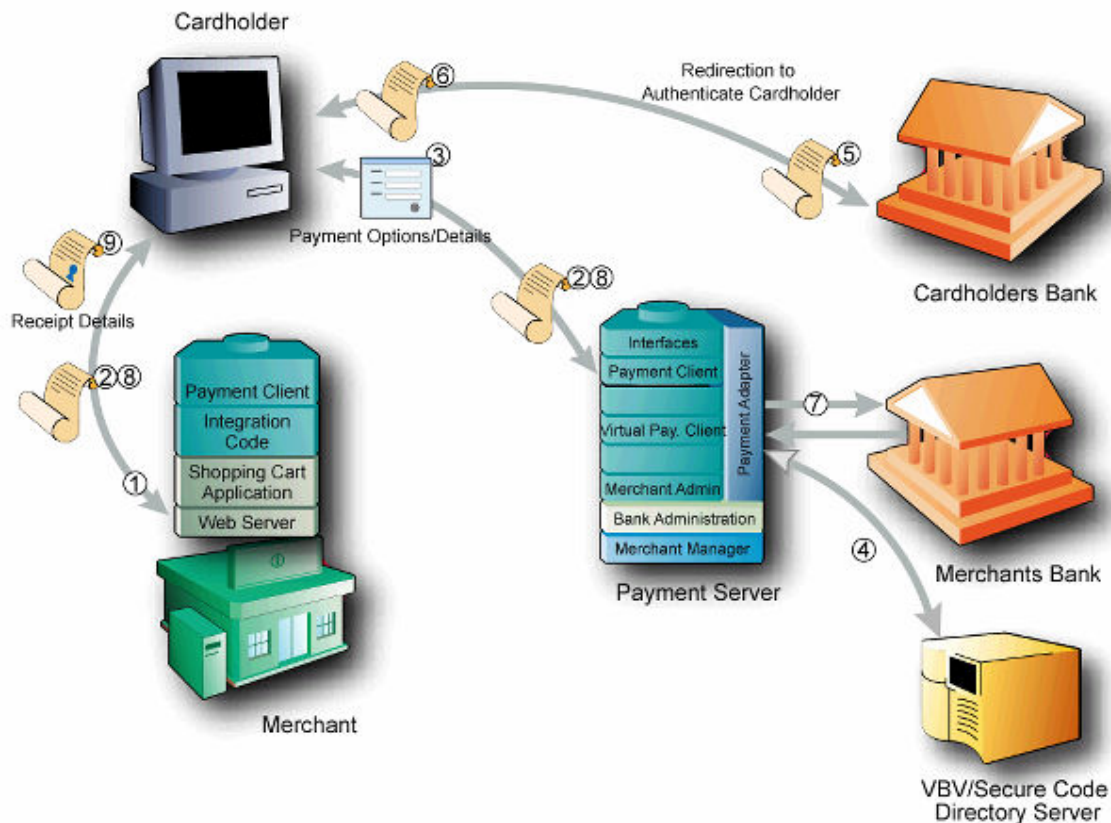
---

## CSC Response Code

As part of the effort to authenticate the identity of the Card Holder, in the case where the authorization are successfully authorized, but the CSC Response Code (vpc\_AcqCSCRespCode) indicates anything other than a match (M), you should consider to void or refund the transaction.

Please refer to page 52 for more details.

## Information Flow of a Payment Authentication



If you have been enabled to use Verified by Visa™ (Visa 3-Domain Secure) and MasterCard SecureCode™ (MasterCard 3-Domain Secure), the information flow for Verified by Visa™ (Visa 3-Domain Secure) and MasterCard SecureCode™ (MasterCard 3-Domain Secure) is:

- 1 A cardholder browses the application, selects a product and enters their shipping details into the merchant's application at the checkout page.
- 2 The cardholder clicks a pay button and your application sends the payment encrypted Transaction Request to the Payment Server by redirecting the cardholder's Internet browser to the Payment Server.
- 3 The Payment Server prompts the cardholder for the card details.
- 4 If the card is a Visa or MasterCard, the Payment Server then checks with the VBV or SecureCode Directory Server to determine if the card is enrolled in either the Verified by Visa™ (Visa 3-Domain Secure) or MasterCard SecureCode™ (MasterCard 3-Domain Secure) scheme.

If the card is not enrolled in payment authentication scheme then the Payment Server continues processing the transaction as a normal 3-Party transaction without payment authentication.

If the cardholder's card is registered in the payment authentication scheme, the Payment Server redirects the cardholder's browser to the card issuing bank site for authentication.

- 5 If the cardholder's card is registered in the payment authentication scheme, the Payment Server redirects the cardholder's browser to the card issuer's site for authentication. The card issuer's server displays the cardholder's secret message and the cardholder enters their secret password, which is checked against the Issuing bank's database.

- 6 At the completion of the authentication stage, the cardholder is redirected back to the Payment Server indicating whether or not the cardholder's password matched the password in the database.  
If the cardholder was not authenticated correctly, then the payment does not take place and the cardholder is redirected back to the merchant's site with an encrypted Transaction Response containing details to indicate the authentication failed - see step 8.
- 7 If the cardholder was authenticated correctly, the Payment Server continues with processing the transaction with the details of the successful authentication operation.
- 8 The Payment Server then redirects the cardholder back to merchant's site with the Transaction Response. The Transaction Response contains the result of the transaction.
- 9 The application processes the Transaction Response and displays the receipt and confirms the order to the cardholder for their records.

Note: If the cardholder is enrolled in the 3D Secure scheme but **is not authenticated correctly**, for example, because the cardholder may have entered their password incorrectly 3 times, then the merchant's application is sent a `vpc_TxnResponseCode` of 'F' to indicate the cardholder failed the authentication process and the transaction does not proceed.

In the case where the card is not found in the card range cache, the Payment Server will return the ECI of the transaction in the Transaction Response field `vpc_3DSECI`.

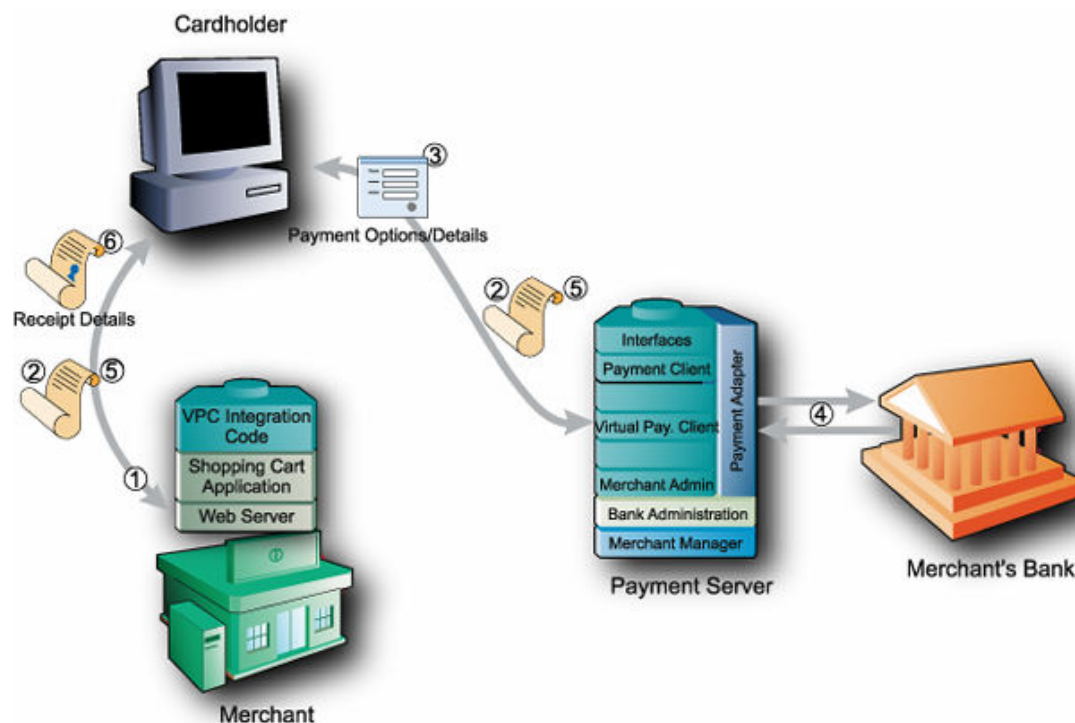


## CHAPTER 6

# 3-Party Payments

This section describes the information flow and integration model for 3-Party Payments.

## Purchase Information Flow in a 3-Party Payment



The following is the information flow in a purchase transaction:

- 1 A cardholder browses your online store, selects a product and enters their shipping details into the merchant's online store at the checkout page.
- 2 The cardholder clicks a pay button and your online store sends the payment request to the Payment Server by redirecting the cardholder's Internet browser to the Payment Server.
- 3 The Payment Server prompts the cardholder for the card details using a series of screens.
  - The first screen displays the cards supported, for example MasterCard, Visa, and American Express. The cardholder chooses the card type they want to use for the transaction.
  - The second screen accepts the details for the chosen card such as card number, card expiry, and card security number if required.
- 4 The Payment Server passes the details to the acquirer bank to process the transaction. After processing, the Payment Server displays the result of the transaction with a receipt number if it was successful or an appropriate information message if it was declined. It then advises the cardholder to wait while they are redirected back to the merchant's site.

- 5 The Payment Server then redirects the cardholder back to merchant's site with the transaction response. The response contains the result of the transaction.
- 6 The online store interprets the response and displays the receipt and confirms the order to the cardholder for their records.

## What the Cardholder Sees

In 3-Party Payments, the cardholder is presented with the following pages:

- 1 The merchant's application checkout page

**UNIVERSAL STORE INVOICE**

Invoice Number: 2760  
Date of purchase: 10/05/01

**YOUR PURCHASES:**

QTY	Code	Brand	Description	Size	Unit Price	Total Price
1	Y224501	SPLIT	JEROME	Small	\$69.95	\$69.95
1	FREIGHT		Express Air Freight		\$20.00	\$20.00
<b>Total:</b>						<b>\$89.95</b>

**ENTER YOUR DELIVERY ADDRESS**

Name: Net Citizen  
 Delivery Address: 2790 Mercy Lane  
 Delivery City: Los Gatos  
 Delivery State: CALIFORNIA  
 Zipcode: 1234  
 Country: United States

**Pay**


The application checkout page displays the line items that the cardholder wants to purchase and the total amount to pay, including any delivery charges and taxes. The cardholder accepts the amount and proceeds to the payment server payment pages to enter their card details. The application checkout page is created by the merchant and displayed on their website.

- 2 The Payment Server's payment options page

The Payment Server creates this and the following pages.




**TEST MODE**




Merchant name: VBV Test profile- QSI



**Select your preferred payment method**


Pay securely using SSL+ by clicking on the card logo below:

[Cancel](#)

Copyright ©2003 Dialect Solutions Holdings Pty Ltd. All Rights Reserved.


SECURE PAYMENTS  POWERED BY DIALECT

The payment options page presents the cardholder with the card types the merchant accepts. The cardholder clicks a card type and proceeds to the Payment Details web page.


3 The Payment Server's payment details page appears.

**TEST MODE**

Merchant name: VBV Test profile- QSI




**Enter your card details**

 **MasterCard:** You have chosen **MasterCard** as your method of payment. Please enter your card details into the form below and click "pay" to complete your purchase.


Card Number  -  -  -

Expiry Date  /  month/year

Security Code  the 3 digits after the card number on the signature panel of your card.




Purchase Amount  **AUD \$123.00**

[Cancel](#) 

I hereby authorise the debit to my MasterCard Account in favour of VBV Test profile- QSI

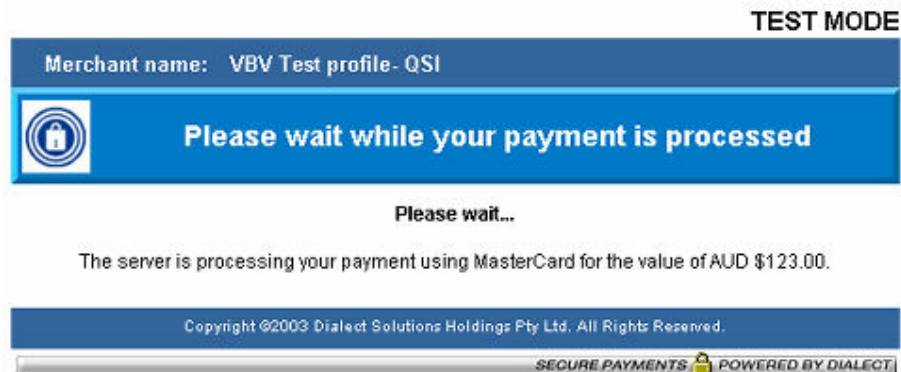
Copyright ©2003 Dialect Solutions Holdings Pty Ltd. All Rights Reserved.

SECURE PAYMENTS  POWERED BY DIALECT

On the Payment Details page, the cardholder enters their card details including the card number, expiry date and card security number (if applicable). Then the Payment Server processes the payment.

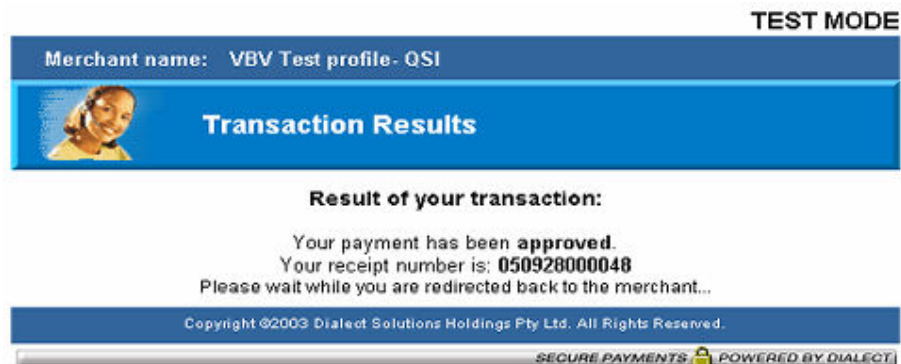
#### 4 The Payment Server's payment pending page

As the bank is processing the payment, a payment pending page can be displayed to the cardholder.

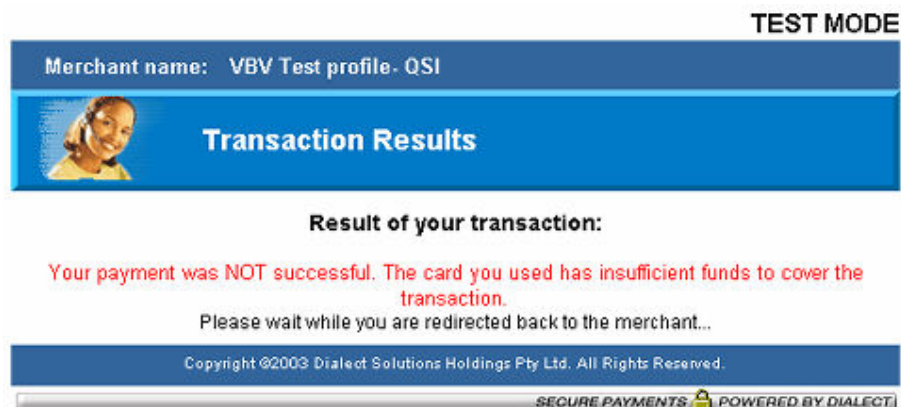


#### 5 The Payment Server's redirection page

The redirection page is displayed in the cardholder's browser and the Transaction Response is passed to your application. A successful transaction would appear as follows:



or, if declined, the following page would appear:



## 6 The merchants application receipt page.

**UNIVERSAL STORE RECEIPT**  
Print a copy of this receipt for your future reference. [Print Receipt](#)

**APPROVED** ✓ **Your purchase payment was approved!**  
We now need the delivery address for your purchase. Simply complete the form below and press "continue" to finalize your purchase.

Name: \_\_\_\_\_ Invoice Number: 2769  
 Delivery Address: \_\_\_\_\_ Date of purchase: 16/05/01  
 City: \_\_\_\_\_  
 State: CALIFORNIA  
 Postcode: 1234  
 Country: United States

**YOUR PURCHASES**

QTY	Code	Brand	Description	Size	Unit Price	Total Price
1	Y234501	S.P.L.I.T.	Jerome	Small	\$69.95	\$69.95
1	FREIGHT		Express Air Freight			\$20.00
<b>Total:</b>						<b>\$89.95</b>

[Continue](#)

The application receives and processes the Transaction Response and displays a receipt page. The application receipt page is created by you and displayed on your website.

## Integrating 3-Party Payments

To process a payment your application needs to be integrated with the Virtual Payment Client in order to create and send the Transaction Request and decrypt the encrypted Transaction Response.

To do this you need to do the following:

### Handle a Digital Order

- 1 Add any session variables required by the application to resume the order process with the cardholder before the DO has been processed. These should be appended to the ReturnURL field.
- 2 Collect the minimum required information for a DO. This includes your merchant Id, an order Information field, the transaction amount, the locale and the return URL where the Payment Server needs to redirect the cardholder back to, and an optional vpc\_MerchTxnRef number if you are using the QueryDR function. You may require additional information fields when using optional features.
- 3 Formulate a DO using the fields outlined above. An example of a DO is:  

```
https://203.202.39.43/pay?5FVersion=1&5FLocale=en&5FCommand=pay&5FAccessCode=A53853CE&5FMerchTxnRef=123&5FMerchant=TESTANDREW&5FOrderInfo=Example&5FAmount=100&5FReturnURL=http%3A%2F%2Flocalhost%2FASP%5FVPC%5F3Party%5FDR%2Easp&
```
- 4 Redirect the cardholder's Internet browser using the DO you just created. At this point the cardholder session with your application is interrupted while the cardholder is redirected to the Payment Server.

## What the Payment Server does

When a Transaction Request arrives at the Payment Server, it:

- Checks to make sure the Transaction Request Digital Signature is correct and if correct and also checks if the optional secure hash is present, if both are correct, the Payment Server:
- and also checks if the optional secure hash is present, if both are correct, the Payment Server:
- and also checks if the optional secure hash is present, if
  - Displays the card selection page for the cardholder to choose their card type.
  - Displays the card details so the cardholder can provide the card details for the selected card type.
  - Processes the data and notifies the acquiring bank of the status of the transaction so the funds can be settled into the merchant's account.
  - Sends back an encrypted Transaction Response to your website page nominated by the ReturnURL in the Transaction Request indicating whether the transaction was successful or declined.
  - The Payment Server can also send error messages back, if for example there is a communication error in the banking network and the transaction cannot proceed.
- If the Transaction Request Digital Signature or the Secure Hash is incorrect, the Payment Server:
  - Displays an error to the cardholder.

## Handle a Digital Receipt

The DR (transaction response) is returned to your website using an Internet browser redirect as specified in the vpc\_ReturnURL field. The DR will always have a secure hash for the online store to check data integrity. An example of a DR is:

```
http://localhost/ASP_VPC_3Party_DR.asp?&vpc_AVSErrorCode=Unsupported&vpc_AcqAVSRespCode=Unsupported&vpc_AcqCSCRespCode=Unsupported&vpc_AcqResponseCode=00&vpc_Amount=100&vpc_AuthorizeId=000281&vpc_BatchNo=1&vpc_CSCRequestCode=N&vpc_CSCResultCode=Unsupported
```

The application receipt page needs to be able to handle:

- Successful transactions
- Declined transactions
- Error Conditions - if vpc\_TxnResponseCode equals '7' or '8' an error has occurred.

All three of these conditions are valid responses that occur back from the Virtual Payment Client.

## Handle Session Variables

A session begins when a cardholder enters your website and ends when they leave your website.

Some merchant applications use session variables to keep track of where the merchant application is up to and to prevent unauthorised entry without the cardholder signing in. This stops hackers from spoofing transactions.

Other applications create session variables in other ways. Some applications don't create session variables at all. If there are no session variable(s) in your application then the next section will not apply.

## Sending Session Variables to the Payment Server

When using 3-Party Payments, the Virtual Payment Client requires the cardholder's browser to support cookies. In 3-Party Payments, the cardholder browser's connection is completely severed from the merchant's application.

Session variables that are required to identify the users must be collected and sent to the Payment Server. The session variables are not used by the Payment Server, but are returned appended to the Transaction Response.

You can store up to 5 session variables using any name that your application needs, providing:

- They conform to HTTP/HTTPS protocols. To make them conform to the standard URL, you need to URL encode all session variables before sending them.
- A URL can only be a maximum total of 2047 characters long, otherwise the browser will not perform a redirect function.
- Their names must not start with **vpc\_**. These variables must be present in the Transaction Request before the MD5 signature is calculated and appended to it.

The session variables are not stored in the Payment Server database. The Virtual Payment Client will send these session fields back to the merchant application in the Transaction Response. This allows the merchant application to recover the session variables from the Transaction Response, and use them to restore the session. The session then continues as though it had never been broken.

---

## Additional 3 Party Functionality

The Payment Server provides you with additional ways to process payments, for example:

- 3-Party Payments, where the merchant collects the cardholder's card type
- 3-Party Payments, where the merchant collects all the cardholder's card details
- 3-Party Payments, where the merchant is enabled for Verified by Visa™ (Visa 3-Domain Secure) and MasterCard SecureCode™ (MasterCard 3-Domain Secure). For more information, see **Using 3-D Secure Authentication** (see "Using 3-D Secure Payment Authentications" on page 29).

## 3-Party Payments where the merchant collects the cardholder's card type

In 3-Party Payments you can choose to bypass the Payment Server payments page that displays the logos of all the cards the Payment Provider will accept. This can be helpful if your application already allows the cardholder to select the card they want to pay with. This stops the cardholder having to do a double selection, once at your application and once on the Payment Server.

You can achieve this by providing the following extra fields in the Transaction Request - Gateway and Card Type. This type of 3-Party transaction is called External Payment Selection (EPS).

## 3-Party Payments where the merchant collects all the cardholder's card details

In 3-Party Payments you can choose to bypass all the Payment Server payment pages displayed. This can be helpful if you want to keep merchant branding consistent throughout a 3-Party transaction. The same security measure must be observed, such as installing an SSL certificate to protect the card details being sent from the cardholder's browser to the merchant's site as in a 2-Party transaction.

You can achieve this by providing the following card details in extra fields in the Transaction Request. These fields are: Card Number, Card Expiry, Gateway and Card Type. You can also submit Card Security Code and any other optional data at this point.

## 3-Party Payments using Verified-by-Visa and/or MasterCard Secure Code

In 3-Party Payments you can have your Payment Provider enable you to use Verified by Visa™ (Visa 3-Domain Secure) and MasterCard SecureCode™ (MasterCard 3-Domain Secure), which provides additional security to all your payments. This type of transaction requires the 3-Party model, and works by redirecting the cardholder to their card issuer to enter a password. For more information, please see Using Verified-by-Visa™ and MasterCard SecureCode™ Payment Authentications to secure your payments (see **Using 3-D Secure Authentication** (see "Using 3-D Secure Payment Authentications" on page 29)).

You can also choose with this transaction type to bypass all the Payment Server payment pages displayed. This can be helpful if you want to keep merchant branding consistent throughout a 3-Party Verified by Visa™ or MasterCard SecureCode™ transaction, except for the one page where the cardholder types in their password.

The same security measure must be observed, such as installing an SSL certificate to protect the card details being sent from the cardholder's browser to the merchant's site as in a 2-Party transaction.

You can achieve this by providing the following card details in extra fields in the Transaction Request. These fields are: Card Number, Card Expiry, Gateway and Card Type. You can also submit Card Security Code and any other optional data at this point.

## CHAPTER 7

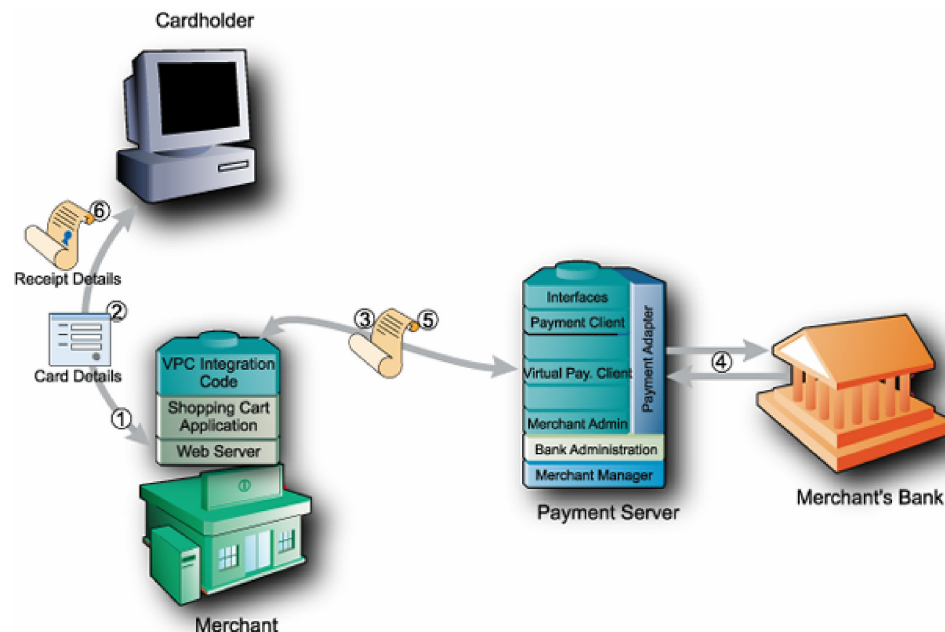
## 2-Party Payments

In the 2-Party Payments Integration Model, a cardholder places an order and provides their card details (card type, card number and expiry date) to you by **Mail Order** or by **Telephone Order (MOTO)** transactions) including Interactive Voice Response (IVR) systems, or some card present application like a ticketing system.

You can implement the 2-Party Payments Integration Model if you prefer cardholders to provide their card details (card type, card number and expiry date) to you rather than to the Payment Server.

2-Party Payments carry a higher risk than 3-Party Payments, as you are responsible for protecting the cardholder's card details.

## 2-Party Payments Information Flow



- 1 A cardholder decides to make a purchase and provides their card details directly to your online store.
- 2 Your application collects the details of the cardholders order.
- 3 In addition it formulates the Transaction Request and sends it via a HTTPS POST over the Internet to the Payment Server via the Virtual Payment Client.
- 4 The Payment Server passes the transaction to the merchant's acquirer bank for processing.
- 5 After processing, the Payment Server generates a Transaction Response and passes it via the Virtual Payment Client to your online store. The Transaction Response shows whether the transaction was successful or not. The results can be stored by you for future reference.

- 6 A receipt is generated and either immediately passed to the cardholder or included when shipping the goods.

## What the Cardholder Sees

In 2-Party Payments over the Internet the cardholder is presented with two pages:

- 1 The merchants application checkout page.
- 2 The merchants application receipt page.

Note: Although you can implement 2-Party Payments with applications other than web stores, an Internet connection is still required to interact with the Payment Server.

---

## CHAPTER 8

---

# Advanced Merchant Administration (AMA)

There are a number of additional transactional options that you can implement, depending on your implementation of Virtual Payment Client. All of these transactions operate using the 2-Party model.

Merchants and users who need AMA transactions must have a username, password and be set up with the appropriate AMA privileges to run a particular AMA transaction.

## AMA Capture

The AMA Capture command allows a merchant to capture the funds from a previous authorisation transaction.

A merchant that operates using Authorisation/Capture mode performs two transactions to transfer the funds into their account.

- 1 The first transaction (Authorisation) reserves the funds on the cardholder's credit card.
- 2 The second transaction (Capture) transfers the funds from the cardholder's account to the merchant's account.

Capture allows a merchant to complete a transaction performed using the Authorisation/Capture Payment Model. The capture transaction initiates the transfer of funds from the cardholder's account to the merchant's account.

**Note:** In Purchase mode, the authorisation and capture operations are completed at the same time in the one purchase transaction, so you do not need to perform a separate capture is not needed, that is **this capture command is not necessary if the merchant is operating in Purchase mode.**

There are two ways you can capture the funds from an authorisation transaction:

1. Manually using Merchant Administration. This is the simplest method if you don't have many transactions. For more information please refer to your Merchant Administration User Guide.
2. Using the Capture command via the Virtual Payment Client to directly perform the capture transaction from your application..

Payment Providers allow merchants to perform as many capture transactions on the original Authorisation transaction as required, but the total amount captured cannot be more than the amount specified in the original Authorisation transaction.

## AMA Refund

AMA Refund allows you to refund funds for a previous purchase or capture transaction from the merchant's account back to the cardholder's account.

Refunds can only be performed for a previously completed a purchase or capture transaction for the particular order. The merchant can run any number of refund transactions on the original transaction, but cannot refund more than has been obtained via a purchase or capture transaction

There are two ways to refund the funds:

- 1 Manually using Merchant Administration. This is the simplest method if the merchant does not have many refund transactions. For more information please refer to your Merchant Administration User Guide.
- 2 Using the AMA Refund command via the Virtual Payment Client to directly perform refunds from the merchant's application.

## AMA Void Capture

AMA Void Capture allows a merchant to void the funds from a previous capture transaction in Auth/Capture mode, that has not been processed by the acquiring institution.

This command cannot be used if the merchant is operating in Purchase mode.

The merchant can only run one void capture transaction on the original capture transaction, as it completely removes the capture transaction as though it never occurred. A void capture must be run before the batch containing the original capture transaction is processed by the acquiring institution.

There are two ways you can Void Capture the funds:

- 1 Manually using Merchant Administration. This is the simplest method if you don't have many Void Capture transactions. For more information please refer to your Merchant Administration User Guide.
- 2 Using the Void Capture command via the Virtual Payment Client to directly perform Void Captures from your application. The merchant must have a user enabled with AMA and Void privileges to use this functionality.

**Note:** Not all financial institutions support void transactions, only those that operate in switch to issuer mode. Please consult with your financial institution if they support voids.

**Note:** Only the most recent transaction in an order can be voided.

## AMA Void Refund

AMA Void Refund allows a merchant to void a previous refund transaction that has not been processed by the acquiring institution.

The merchant can only run one Void Refund transaction on the original refund transaction as it completely removes the refund transaction as though it never occurred. The Void Refund must be run before the acquiring institution processes the batch containing the original refund transaction.

There are two ways you can Void Refund the funds. Your Payment Provider must enable this function on your Merchant Profile for you to use either of these methods:

- 1 Manually using Merchant Administration. This is the simplest method if you don't have many Void Refund transactions. For more information please refer to your Merchant Administration User Guide.
- 2 Using the Void Refund command via the Virtual Payment Client to directly perform Void Refund from your application. The merchant must have a user enabled with AMA and Void privileges to use this functionality.

**Note:** Not all financial institutions support void transactions. Please consult with your financial institution if they support voids.

**Note:** Only the most recent transaction in an order can be voided.

## AMA Void Purchase

AMA Void Purchase allows a purchase merchant to void a purchase transaction that has not been processed by the acquiring institution. It is not available for Auth/Capture mode merchants. This transaction is not possible for Debit and EBT transactions.

The merchant can only run one 'Void Purchase' transaction on the original 'Purchase' transaction as it completely removes the purchase transaction as though it never occurred.

The Admin Void Purchase must be run before the acquiring institution processes the batch containing the original purchase transaction.

There are two ways you can Void Purchase the funds. Your Payment Provider must enable this function on your Merchant Profile for you to use either of these methods:

- 1 Manually using Merchant Administration. This is the simplest method if you don't have many Void Purchase transactions. For more information please refer to your Merchant Administration User Guide.
- 2 Using the Void Purchase command via the Virtual Payment Client to directly perform Void Purchase from your application. The merchant must have a user enabled with AMA and Void privileges to use this functionality.

**Note:** Not all banks support void transactions, only those that operate in switch to issuer mode. Please consult with your financial institution if they support voids.

**Note:** Only the most recent transaction in an order can be voided.



## CHAPTER 9

# Basic Transactions API Reference Fields

These APIs enable you carry out basic transactions correctly.

## 3-Party Payment Transactions

3-Party Payments requires you to use <https://migs.mastercard.com.au/vpcpay> for the Virtual Payment Client.

**Note:** You must use HTTPS protocol or the Virtual Payment Client will reject the DO.

### DO Fields for 3-Party Payment Requests

Transaction Requests contain the information collected for a cardholder's order that is used for processing by the Payment Server.

The transaction request must include all the required fields for 3-Party Payments. You can also include optional fields such as Card Security Code.

#### Required DO fields for a 3-Party Payment Request

The required fields that must be included in a DO when using 3-Party Payments are:

Field Name	Required Optional	Field Type	Length	Example Value
vpc_Version	The version of the Virtual Payment Client API being used. The current version is 1.			
	Required	Alphanumeric	1,8	1
vpc_Command	Indicates the transaction type. This must be equal to <b>pay</b> .			
	Required	Alphanumeric	1,16	pay
		<p>A unique value created by the merchant to identify the DO. It is used to track the progress of a transaction and allows it to be identified on the Payment Server should a communication's failure occur and the DR is not received. It can contain similar information to the <i>vpc_OrderInfo</i> field, but it must be unique.</p> <p>It may be in part an order number or invoice number, but it should also reflect the transaction attempt. For example, if a cardholder has insufficient funds on their card and you allow them to repeat the transaction with another credit card. The value may be test1234/1 on the first attempt, test1234/2 on the second attempt and test1234/3 on the third attempt.</p> <p>It can use text made up of any of the base US ASCII characters in the range, hexadecimal 20 to 126.</p>		

Field Name	Required Optional	Field Type	Length	Example Value
	Required	Alphanumeric - Special characters	1,40	test1234/1
vpc_AccessCode	The access code authenticates you on the Payment Server so that a merchant cannot access another merchant's MerchantId. The access code is provided to you when you registered your merchant profile with your Payment Provider.			
	Required	Alphanumeric	8	6ab89f3
vpc_Merchant	The unique merchant Id assigned to you by your Payment Provider.			
	Required	Alphanumeric	1,16	TESTMERCHANT01
vpc_OrderInfo	Your own identifier used to identify the transaction with the cardholder. For example, a shopping cart number, an order number, or an invoice number.			
	Optional	Alphanumeric - Special characters	1,34	test1234
vpc_Amount	The amount of the transaction in the smallest currency unit expressed as an integer. For example, if the transaction amount is \$49.95 then the amount in cents is 4995.			
	Required	Numeric	1,10	4995
vpc_Locale	Used in SSL type transactions for specifying the language that is used on the Payment Server pages that are displayed to the cardholder. If the locale is not supplied the Payment Server defined default of 'en' is used.			
	Required	Alphanumeric	2,5	en_AU
vpc_ReturnURL	The URL that is displayed to the cardholder's browser when the Payment Server sends the DR. It must be a complete URL. The Return URL must start with either http:// or https:// and may be up to 255 characters. If the return URL is not supplied, your default vpc_ReturnURL that you nominated when you registered your merchant profile with your Payment Provider is used.			
	Required	Alphanumeric - Special characters	1,255	http://returnurl/Receipt.asp

### Optional DO fields for a 3-Party Payment Request

The optional fields that can be included in a DO when using 3-Party Payments are:

Field Name	Required Optional	Field Type	Length	Example Value
Secure Hash				
vpc_SecureHash	Used to allow the Virtual Payment Client to check the integrity of the DO. <b>Store Secure Hash securely</b> (see "Store Secure Hash Secret Securely" on page 29)			
	Optional	Alphanumeric	32	68798ab0259eb01be7bbe 2a807171f83
Verified by Visa™ (Visa 3-Domain Secure) and MasterCard SecureCode™ (MasterCard 3-Domain Secure) - Optional DO Fields				

Field Name	Required Optional	Field Type	Length	Example Value
Verified by Visa™ (Visa 3-Domain Secure) and MasterCard SecureCode™ (MasterCard 3-Domain Secure) are payment authentications designed to prevent credit card fraud by redirecting cardholders to their card issuer where they enter a password that they had previously registered with their card issuer. No additional input fields are required for a Verified-by-Visa or MasterCard Secure Code payment authentication, but you do need your Payment Provider to enable you to use Verified by Visa™ (Visa 3-Domain Secure) or MasterCard SecureCode™ (MasterCard 3-Domain Secure).				
Ticket Number - Optional DO Fields				
vpc_TicketNo	Allows you to include a ticket number, such as an airline ticket number in the DO. The ticket number is stored on the Payment Server database for the transaction. The ticket number is not returned in the DR.			
	Optional	Alphanumeric - Special characters	1,15	AB1234

## Optional Merchant Defined Fields for 3-Party Payment Requests

3-Party Payments also supports up to 5 merchant defined fields that will be returned to you in the DR. These fields must be less than 255 bytes and must not start with vpc\_. These fields are not stored in the Payment Server. For more information, please see **Session Variables** (see "Handle Session Variables" on page 39).

## Sending a DO for 3-Party Payments

Usually the GET method with a Query String containing the transaction request fields, and a HTTPS Redirect, is used to send the transaction request via the Virtual Payment Client to the Payment Server, when using 3-Party Payments.

However if the 3-Party Payment DO is sent with Card Details, a POST method should be used to send the transaction request. Please refer to the relevant sample codes for examples.

## DR Fields for 3-Party Payments

### Required DR fields for 3-Party Payment Response

The DR contains the results of the DO fields that were processed by the Payment Server. It indicates whether the payment was successful or not.

The required fields that are included in the DR fields for 3-Party Payments are:

Field Name	Required Optional	Field Type	Length	Example Value
3-Party Payments - DR fields				
vpc_Version	The value of the <i>vpc_Version</i> DO input field that is returned in the DR.			
	Input	Alphanumeric	1,2	1
vpc_Command	The value of the <i>vpc_Command</i> DO input field that is returned in the DR.			
	Input	Alpha	3	pay

Field Name	Required Optional	Field Type	Length	Example Value
<i>vpc_MerchTxnRef</i>	The value of the <i>vpc_MerchTxnRef</i> DO input field that is returned in the DR.			
	Input	Alphanumeric - Special characters	1,40	test1234/1
<i>vpc_Merchant</i>	The value of the <i>vpc_Merchant</i> DO input field that is returned in the DR.			
	Input	Alphanumeric - Special characters	1,16	TESTMERCHANT01
<i>vpc_OrderInfo</i>	The value of the <i>vpc_OrderInfo</i> DO input field that is returned in the DR.			
	Input	Alphanumeric - Special characters	1,34	test1234
<i>vpc_Amount</i>	The value of the <i>vpc_Amount</i> DO input field that is returned in the DR.			
	Input	Numeric	1,10	4995
<i>vpc_Locale</i>	The value of the <i>vpc_Locale</i> DO input field that is returned in the DR. It specifies the language that is used on the Payment Server pages that are displayed to the cardholder. If the Locale is not supplied in the DO, the default value of 'en' (English) used in the Payment Server.			
	Input	Alphanumeric	2,5	en_AU
<i>vpc_TxnResponseCode</i>	A response code that is generated by the Payment Server to indicate the status of the transaction. A <i>vpc_TxnResponseCode</i> of "0" (zero) indicates that the transaction was processed successfully and approved by the Acquiring Bank. Any other value indicates the transaction was declined. See Response Code.			
	Required	Alphanumeric	1	0
<i>vpc_transactionNo</i>	A unique number generated by the Payment Server for the transaction. It is stored in the Payment Server as a reference and used to perform actions such as a refund or capture.			
	Required	Numeric	1,12	3465
<i>vpc_Message</i>	Indicates any errors the transaction may have encountered.			
	Optional	Alphanumeric	10,200	Merchant [TESTCORE23] does not exist
<i>vpc_AcqResponseCode</i>	Acquirer's Response Code is generated by the financial institution to indicate the status of the transaction. The results can vary between institutions so it is advisable to use the <i>vpc_TxnResponseCode</i> as it is consistent across all acquirers. It is only included for fault finding purposes.			
	Optional	Alphanumeric	2,3	00
<i>vpc_ReceiptNo</i>	This is also known as the Reference Retrieval Number (RRN), which is a unique identifier. This value is passed back to the cardholder for their records if the merchant application does not generate its own receipt number.			
	Optional	Alphanumeric	1,12	RP12345

Field Name	Required Optional	Field Type	Length	Example Value
	Input			
vpc_BatchNo		A date supplied by the acquirer to indicate when this transaction will be settled. If the batch has today's date then it will be settled the next day. When the acquirer closes the batch at the end of the day, the date will roll over to the next processing day's date.		
	Optional	Alphanumeric	1,8	20021021
vpc_AuthorizeId		An identifying code issued by the bank to approve or deny the transaction. This is an optional field and may not be supplied by all acquirers.		
	Optional	Alphanumeric	1,12	ABC12345
vpc_Card		A code issued by the Payment Server for the card type used by the cardholder in the transaction. For a list of card type codes, see <b>Card Type Code</b> (see "Card Type Code" on page 91) .		
	Optional	Alphanumeric	2	ABC12345

**Optional DR fields for 3-Party Payment Response**

If you integrate advanced functionality when using 3-Party Payments, then optional fields that can be included in a DR from the Payment Server are:

Field Name	Required Optional	Field Type	Length	Example Value
	Input			
<b>Secure Hash Secret - DR field</b>				
vpc_SecureHash		This field is only returned for a 3-Party Payment as the response is returned via the cardholder's browser as a QueryString, which is visible to the cardholder. It allows you to check message integrity to ensure the response values have not been tampered with. Secure hash is only optional if you have, the mayOmitHash privilege provided to you by your Payment Provider. For more information see <b>Secure Hash Secret</b> (see "Store Secure Hash Secret Securely" on page 29).		
	Optional	Alphanumeric	32	68798ab0259eb01be7bbe2a807171f83
<b>Card Security Code (CSC) - DR fields</b>				

Field Name	Required Optional	Field Type	Length	Example Value
vpc_AcqCSCRespCode	Input			<p>The response code generated by the Issuing Bank in relation to the Card Security Code:</p> <p><b>For MasterCard Transactions:</b></p> <p>M: Match                      N: Not Match                      P: Unable to Process                      U: Issuer unregistered to process CVC 2/CVV2                      X: Reserved for Magneprint</p> <p><b>For Visa Transactions</b></p> <p>M: Match                      N: Not Match                      P: Unable to Process                      S: CVV2 is on card but the merchant has indicated that CVV2 is not present                      U: Issuer is not Visa-certified for CVV2, has not provided VISA encryption keys or both.</p> <p><b>For American Express:</b></p> <p>Currently, the CID is not supported for Amex transactions. However the Amex CID is indeed passed on for verification, the result is conclusive in the Transaction Response Code (vpc_TxnResponseCode).</p>
vpc_CSCResultCode	Optional	Alphanumeric	1	S
	Optional	Alphanumeric	1	M

Verified by Visa™ (Visa 3-Domain Secure) and MasterCard SecureCode™ (MasterCard 3-Domain Secure) - **DR fields**

Field Name	Required Optional	Field Type	Length	Example Value
<p>These fields are only returned in the DR if the transaction is a Verified by Visa™ (Visa 3-Domain Secure) and MasterCard SecureCode™ (MasterCard 3-Domain Secure) payment authentication. You must be enabled on the Payment Server by your Payment Provider to perform Verified by Visa™ (Visa 3-Domain Secure) and MasterCard SecureCode™ (MasterCard 3-Domain Secure) payment authentications.</p> <p>The <code>vpc_TxnResponseCode</code> is used to determine if the authentication passed or a failed.</p> <p>If the <code>vpc_TxnResponseCode</code> is not equal to 'F', the payment authentication passed OK and the Authentication process has completed satisfactorily.</p> <p>If the <code>vpc_TxnResponseCode</code> is equal to 'F', the Authentication process failed and no payment took place.</p> <p>If a payment authentication has been successful, extra fields are returned in the DR for a Verified-by-Visa and MasterCard Secure Code payment authentication. The fields are not used by you but are returned to allow you to store them as a record of authentication for the transaction, which can be used to resolve disputes. They cannot be used again for any future transactions.</p> <p>All payment authentication transactions use a <code>vpc_VerStatus</code> response code value to show whether the card authentication was successful or not. For details of this code, please see <b>Verified by Visa™ (Visa 3-Domain Secure) and MasterCard SecureCode™ (MasterCard 3-Domain Secure) Status Codes</b> (see "Verified by Visa™ (Visa 3-Domain Secure) and MasterCard SecureCode™ (MasterCard 3-Domain Secure) Status Codes" on page 92).</p>				
vpc_VerType	Either '3DS' or 'SPA'.			
	Optional	Alphanumeric	3,20	3DS
vpc_VerStatus	The status codes used by the Payment Server - see page 83.			
	Optional	Alphanumeric	1	N
vpc_VerSecurityLevel	<p>The Verification Security Level is generated at the card issuer as a token to prove that the cardholder was enrolled and authenticated OK. It is shown for all transactions except those with authentication status "Failure". This field contains the security level to be used in the AUTH message.</p> <p>'05' - Fully Authenticated.</p> <p>'06' - Not authenticated, (cardholder not participating), liability shift.</p> <p>'07' - Not authenticated. Usually due to a system problem, for example the merchant password is invalid.</p>			
	Optional	Numeric	1,2	06
vpc_VerToken	This value is generated by the card issuer as a token to prove that the cardholder authenticated OK. This is a base64 encoded value.			
	Optional	Alphanumeric	28	glGCg4SFhoeliYqLjI2Oj5CRkpM=
vpc_3DSXID	It is a unique transaction identifier that is generated by the merchant to identify the 3DS transaction. It is a 20-byte field that is Base64 encoded to produce a 28-character value.			
	Optional	Alphanumeric	28	uyPfGIgsoFQhklkIsto+IFWs92s=
vpc_3DSECI	The 3-D Secure Electronic Commerce Indicator, which is set to '05' when the cardholder authenticates OK, and '08' when the cardholder is not enrolled. (These values may change depending on the locale or issuer).			
	Optional	Numeric	2	08

Field Name	Required Optional	Field Type	Length	Example Value
	Input			
vpc_3Dsenrolled	This field is only included if the card is within an enrolled range. This is the value of the VERes.enrolled field. It will take values ( <b>Y</b> - Yes, <b>N</b> - No, <b>U</b> - Unavailable for Checking).			
	Optional	Alpha	1	N
vpc_3Dsstatus	This field is only included if payment authentication was used and a PAREs was received by the MPI. It will take values ( <b>Y</b> - Yes, <b>N</b> - No, <b>A</b> - Attempted Authentication, <b>U</b> - Unavailable for Checking).			
	Optional	Alpha	1	N

### Payment Provider Defined Fields for 3-Party Payments

In addition to the above fields, a further set of optional DR fields may be available, known as *Payment Provider Defined DR Fields*.

Please consult the documentation provided from your Payment Service Provider for more information on the available Payment Provider Defined Fields, their meaning, Field Type and Length.

#### *Receiving the DR*

To receive the DR, you must specify a return Internet address (return /URL). This address is also where the cardholder is returned to when they have paid for their goods/services. The return URL (`vpc_ReturnURL`) is set up when you registered your merchant profile but it can be overridden by setting the `vpc_ReturnURL` field of a DO.

A default value for the `vpc_ReturnURL` field can be stored in the Payment Server. If you only want to use this value, the `vpc_ReturnURL` field must be set to an empty string, "".

If the value passed in the `vpc_ReturnURL` field starts with "http://" or "https://", the contents of the value passed into the DO overwrites the default value stored in the Payment Server.

If the default return URL value is stored in the Payment Server, then the `vpc_ReturnURL` field must contain a valid URL (starting with "http://" or "https://") for every DO.

If the `vpc_ReturnURL` field **does not start** with "http://" or "https://", the contents of the DO is appended to the value in the Payment Server.

If the resultant ReturnURL value does not form a valid URL, an error is generated in the Payment Server which will stop the transaction.

## Calculating and Validating the Secure Hash Secret

Secure Hash Secret is used to detect whether the DO and response has been tampered with. It is added to the DO details before an MD5 algorithm is applied to generate a secure hash. The secure hash is then sent to the Payment Server with the DO details. Because the Payment Server is the only other entity apart from your VPC integration that knows your secure hash secret it recreates the same secure hash and matches it with the one that you sent. If they match the Payment server continues processing the transaction. If it doesn't match, it assumes that the DO has been tampered with and will stop processing the transaction and send back an error message.

For more information on Secure Hash Secret, please see *Detect alteration of DOs and responses using Secure Hash* (see "Additional features for 3-Party Transactions" on page 28).

### How the Secure Hash is Created and Verified

The `vpc_SecureHash` field is used for the MD5 (MD5 RFC1321) secure hash of your secure hash secret and the DO.

The secure hash value is the Hex encoded MD5 output of the DO or response fields. The order that the fields are hashed in are:

- 1 The Secure Hash Secret is always first.
- 2 Then all DO fields are concatenated to the Secure Hash Secret in alphabetical order of the field name. The sort should be in ascending order of the ASCII value of each field string. If one string is an exact substring of another, the smaller string should be before the longer string. For example, Card should come before CardNum.
- 3 Fields must not have any separators between them and must not include any null terminating characters.

For example, if the Secure Hash Secret is 0F5DD14AE2E38C7EBD8814D29CF6F6F0, and the DO includes the following fields:

Field Name	Example Value
<code>vpc_Merchant</code>	MER123
<code>vpc_OrderInfo</code>	A48cvE28
<code>vpc_Amount</code>	2995

In ascending alphabetical order the DO fields inputted to the MD5 hash would be:

```
0F5DD14AE2E38C7EBD8814D29CF6F6F02995MER123A48cvE28
```

### Adding Secure Hash to a DO

Although the risk of a cardholder tampering with the DO is minimal, it is recommended that you include a Secure Hash in your DO. If a cardholder changes a DO, it will be detected because if the Secure Hash generated by the Payment Server does not match the one generated by you, the payment is rejected.

If the secure hash does not match, the Virtual Payment Client will immediately return the cardholder to the merchant's site with an error, by setting the `vpc_TxnResponseCode` field to 7 to indicate that the secure hash is incorrect.

- During integration, this may mean that you have not calculated your hash properly.
- During production, this would usually mean that a cardholder is attempting to commit fraud.
- To create a Secure Hash, the following fields are required for a DO using 3-Party Payments.

```
<input type="hidden" name="vpc_Version" value="1">
<input type="hidden" name="vpc_AccessCode" value="6ab89f3">
<input type="hidden" name="vpc_Merchant" value="TESTWEBNAB01">
```

```
<input type="hidden" name="vpc_OrderInfo" value="test1234">
<input type="hidden" name="vpc_Amount" value="4995">
<input type="hidden" name="vpc_Locale" value="en">
<input type="hidden" name="vpc_ReturnURL"
value="http://192.168.21.205/Receipt.asp">
<input type="hidden" name="Secure_Secret"
value="ebe65403de22d35c7685cb8403315c00">
```

The fields in the DO must be concatenated in ascending alphabetical order with the Secure Hash Secret first:

```
md5_input = Secure_Secret + vpc_AccessCode + vpc_Amount + vpc_Locale + vpc_Merchant
+ vpc_OrderInfo + vpc_ReturnURL + vpc_transactionNo + vpc_Version
```

The order used is the Virtual Payment Client field names, not the alphabetical order of the names you may use in your application.

Any extra functionality fields must be also concatenated to the md5\_input in ascending alphabetical order as shown above, for example, if Ticket Number functionality is added, then the extra field to be added is:

```
<input type="hidden" name="vpc_TicketNo" value="ABC123">
```

And the md5\_input would then become:

```
md5_input = Secure_Secret + vpc_AccessCode + vpc_Amount + vpc_Locale +
vpc_Merchant + vpc_OrderInfo + vpc_ReturnURL + vpc_TicketNo +
vpc_transactionNo + vpc_Version
```

You should also ensure that:

- UTF-8 encoding should be used to convert the input from a printable string to a byte array. Note that 7-bit ASCII encoding is unchanged for UTF-8.
- The hash output must be hex-encoded.

### Adding Secure Hash to a DR

When you receive the DR from the Payment Server, you should calculate the Secure Hash and compare it to the Secure Hash from the Payment Server to ensure that the data has not been tampered with in the DR.

If you do not check the Secure Hash, the DR can be retrieved securely from the Payment Server using QueryDR.

To create a Secure Hash, the following fields are required for a DR using 3-Party Payments.

```
String version      = req.getParameter("vpc_Version ");
String merchantId   = req.getParameter("vpc_Merchant");
String orderInfo    = req.getParameter("vpc_OrderInfo");
String amount       = req.getParameter("vpc_Amount");
String locale       = req.getParameter("vpc_Locale");
String txnResponseCode = req.getParameter("vpc_TxnResponseCode");
String acqResponseCode = req.getParameter("vpc_AcqResponseCode");
String receiptNo    = req.getParameter("vpc_ReceiptNo");
String xtnNo        = req.getParameter("vpc_transactionNo");
String batchNo      = req.getParameter("vpc_BatchNo");
String authorizeID   = req.getParameter("vpc_AuthorizeId");
String resp_Secure_Hash = req.getParameter("vpc_SecureHash");
```

The fields in the DR must be concatenated in ascending alphabetical order with the Secure Hash Secret first:

```
md5_input = Secure_Secret + amount + authorizeID + batchNo + locale + merchantId +
orderInfo + qsiResponseCode + receiptNo + transactionNo + version
```

The order used is the Virtual Payment Client field names, not the alphabetical order of the names you may use in your application.

You should also ensure that:

- UTF-8 encoding should be used to convert the input from a printable string to a byte array. Note that 7-bit ASCII encoding is unchanged for UTF-8.
- The hash output must be hex-encoded.

## 2-Party Payment Transactions

2-Party Payments require you to use `https://<vpc_name>/vpcdps` URL for the Virtual Payment Client, where your Payment Provider will supply the `<vpc_name>` for the URL of the Payment Server.

You must use HTTPS protocol or the Virtual Payment Client will reject the DO.

In 2-Party Payments the application connects directly to the Virtual Payment Client using a form POST operation that directly returns a response. Since the Payment Server cannot collect cardholder card details, they must be collected on your site and sent to the Virtual Payment Client.

During 2-Party Payments, session variables do not need to be sent to the Payment Server because the merchant's session is not broken as it is in 3-Party Payments, where the cardholder's Internet browser is disconnected from the merchant's site and redirected to the Payment Server.

This means that in 2-Party Payments, the cardholder browser is not redirected, so advanced functionality such as Verified-by-Visa and MasterCard Secure Code cannot be used.

## Digital Order Fields for 2-Party Payments

The DO contains the required information for a cardholder's order that is sent via the Virtual Payment Client to the Payment Server.

### Required DO fields for 2-Party Payment Request

The required fields that must be included in a DO when using 2-Party Payments are:

Field Name	Required Optional Input	Field Type	Length	Example Value
vpc_Version	The version of the Payment Client API being used. The current version is 1. Required	Alphanumeric	1,8	1
vpc_Command	Indicates the type of transaction type. It must be equal to <b>'pay'</b>			

Field Name	Required Optional Input	Field Type	Length	Example Value
	Required	Alpha	1,16	pay
<i>vpc_MerchTxnRef</i>	A unique value created by you to identify the DO. It is used to track the progress of a transaction and allows it to be identified on the Payment Server should a communication's failure occur and the DR is not received. It can contain similar information to the <i>vpc_OrderInfo</i> field, but it must be unique. It may be in part an order number or invoice number, but it should also reflect the transaction attempt. For example, if a cardholder has insufficient funds on their card and you allow them to repeat the transaction with another credit card. The value may be test1234/1 on the first attempt, test1234/2 on the second attempt and test1234/3 on the third attempt. It can use text made up of any of the base US ASCII characters in the range, hexadecimal 20 to 126.			
	Required	Alphanumeric - Special characters	1,40	test1234/1
<i>vpc_AccessCode</i>	The access code is used to authenticate you on the Payment Server so that a merchant cannot access another merchant's MerchantId. The access code is provided to you when you registered your merchant profile with the Payment Provider.			
	Required	Alphanumeric	8	6ab89f3
<i>vpc_Merchant</i>	The unique merchant Id assigned to you by your Payment Provider.			
	Required	Alphanumeric	1,16	TESTMERCHANT01
<i>vpc_OrderInfo</i>	An identifier provided by you to identify the transaction with the cardholder. It can be a shopping cart number, an order number, or an invoice number.			
	Optional	Alphanumeric - Special characters	1,34	test1234
<i>vpc_Amount</i>	The amount of the transaction in the smallest currency unit expressed as an integer. For example, if the transaction amount is \$49.95 then the amount in cents is 4995.			
	Required	Numeric	1,10	4995
<i>vpc_CardNum</i>	The number of the card to be used for processing the payment. It can only be a long integer value with no white space or formatting characters.			
	Required	Numeric	15,40	5123456789012346
<i>vpc_CardExp</i>	The expiry date of the card to be processed for payment. The format for this is <b>YYMM</b> , for example, for an expiry date of May 2009, the value would be 0905. The value must be expressed as a 4-digit number (integer) with no white space or formatting characters			
	Required	Numeric	4	0504

## Optional DO fields for 2-Party Payment Requests

The optional fields that can be included in a DO when using 2-Party Payments are:

Field Name	Required Optional	Field Type	Length	Example Value
<b>Card Security Code (CSC) - Optional DO Fields</b>				
vpc_CardSecurityCode				
	The Card Security Code (CSC) is a security feature used for card not present transactions that compares the Card Security Code on the card with the records held in the card issuer's database. For example, on Visa and MasterCard credit cards, it is the three digit value printed on the signature panel on the back following the credit card account number. For American Express, the number is the 4 digit value printed on the front above the credit card account number.			
	Optional	Numeric	1,4	123
<b>Ticket Number - Optional DO Fields</b>				
vpc_TicketNo				
	This allows the merchant to include a ticket number, such as an airline ticket number in the DO. The ticket number is stored on the Payment Server database for that transaction. The ticket number value is not returned in the DR.			
	Optional	Alphanumeric - Special characters	1,15	AB1234
<b>Merchant Transaction Source</b>				
Merchant transaction source functionality allows a merchant to indicate the source of a 2 Party transaction. Merchants and acquirers can optionally set the merchant transaction source so the payment provider can calculate correct fees and charges for each transaction.				
vpc_TxSource				
	Allows the merchant to specify the source of the transaction. This can only be used if the merchant has their privilege set to use this command, otherwise the transaction will be set to the merchant's default transaction source as defined by your Merchant Manager.			
	Optional	Alphanumeric	1,16	INTERNET - indicates an Internet transaction MOTOCC - indicates a call centre transaction MOTO - indicates a mail order or telephone order MAILORDER - indicates a mail order transaction TELORDER - indicates a telephone order transaction CARDPRESENT - indicates that the merchant has sighted the card. VOICERESPONSE - Indicates that the merchant has captured the transaction from an IVR system.

Merchant Transaction Source Subtype				
vpc_TxSourceSubType	Allows the merchant to flag the subtype of transaction for the cardholder's order.			
	Required	Alphanumeric	12	<p>Must be one of the following values:</p> <p><b>SINGLE</b> - indicates a single transaction where a single payment is used to complete the cardholder's order</p> <p><b>RECURRING</b> - indicates a recurring transaction where the cardholder authorises the merchant to automatically debit their accounts for bill or invoice payments. This value only indicates to the acquirer that this is a recurring type payment; it does not mean that the merchant can use the Payment Server's Recurring Payment functionality.</p>

#### CardPresent - Optional DO Fields

This feature allows merchants to add Card Present information and track data to a transaction. This feature applies where the merchant integration collects card track data from POS terminals. Card present functionality can only be performed as a 2-party Authorisation/Purchase transaction. The card track data needs to contain the correct start and end sentinel characters and trailing longitudinal redundancy check (LRC) characters. For all card present transactions, the Merchant Transaction Source, must be set to the value '**CARDPRESENT**'. Regarding card track data,

- If both are available, both *vpc\_CardTrack1* and *vpc\_CardTrack2* must be added to the Transaction Request OR
- If only one is available, either *vpc\_CardTrack1* or *vpc\_CardTrack2* must be added to the Transaction Request.

If the magnetic stripe data is not available, for example, if the card is defective, or the POS terminal was malfunctioning at the time, it is sufficient to set the merchant transaction source to '**CARDPRESENT**' and change the '*PAN Entry Mode*' and '*PIN Entry Capability*' values in *vpc\_POSEntryMode* field to indicate that the card was sighted, but manually entered.

**Note: Track 3 data is not supported.**

#### **vpc\_CardTrack2**

7 bit ASCII text representing the card track 2 data.

Optional	Alphanumeric	38,40	;5123456789012346=13051019681143384001?
----------	--------------	-------	---

#### **vpc\_TxSource**

The source of the transaction.

This must be set to **CARDPRESENT** if the merchant's default transaction source has not been configured to **CARDPRESENT**.

Optional	Alphanumeric	11	CARDPRESENT
----------	--------------	----	-------------

## 2 Party Style Pre-Authenticated Payment Transactions

In a 2 Party Style Pre-Authenticated Payment transaction, the merchant's application can stop the transaction from progressing to the Payment stage and return an error message back to the cardholder if the cardholder is not enrolled in 3-D Secure.

Input fields must be added using `addDigitalOrderField("Fieldname", <field value>)` to the DO for a 2 Party Style Pre-Authenticated Payment transaction mode. The fields contain the results of the authentication.

### DO Fields for 2 Party Style Pre-Authenticated Payment Transactions

The DO Input fields for a 2 Party Style Pre-Authenticated Payment transaction are:

Field Name	Required Optional Input	Field Type	Length	Example Value
vpc_VerType	This field must be a value of '3DS' for the following fields to operate			
	Required	Alphanumeric	3	3DS
vpc_VerToken	This value is generated by the Access Control Server at the card issuer as a token to prove that the cardholder authenticated OK. This is a base64 encoded value.			
	Required	Alphanumeric	28	
vpc_3DSXID	It is a unique transaction identifier that is generated by the merchant to identify the 3DS transaction. It is a 20-byte field that is Base64 encoded to produce a 28-character value.			
	Required	Alphanumeric	28	
vpc_3DSECI	It is the 3-D Secure Electronic Commerce Indicator, which is returned from the Issuers ACS. For VBV, this is '05' where the Issuers ACS has validated the cardholders password or '06' where an 'Attempts ACS' condition has occurred. For Mastercard SecureCode, the value will be either '01' or '02'.			
	Required	Alphanumeric	2	05
vpc_3DSenrolled	This field is mandatory if the card is within an enrolled range. This is the value of the VERes.enrolled field. It will take values ( <b>Y</b> - Yes, <b>N</b> - No, <b>U</b> - Unavailable for Checking).			
	Optional	Alphanumeric	1	Y
vpc_3DSstatus	This field is only included if 3-D Secure authentication was used and a PARes was received by the MPI. It will take values ( <b>Y</b> - Yes, <b>A</b> - Attempted Authentication, <b>U</b> - Unavailable for Checking).			
	Optional	Alphanumeric	1	Y

## Sending a DO for 2-Party Payments

### Sending a DO using the Post Method

The Post Method is used when you collect the cardholder's card details. The data is collected in a secure form and included in the DO that is sent directly to the Payment Server.

The following post method example shows the minimum number of fields required to complete a transaction using the 2-Party Payment integration model:

```
<form method="POST" action="https://www.<vpc_name>/vpcdps">
```

```

<input type="hidden" name="vpc_Version" value="1">
<input type="hidden" name="vpc_Command" value="pay">
<input type="hidden" name="vpc_AccessCode" value="6ab89f3">
<input type="hidden" name="vpc_MerchTxnRef" value="test1234/1">
<input type="hidden" name="vpc_Merchant" value="TESTWEBNAB01">
<input type="hidden" name="vpc_OrderInfo" value="test1234">
<input type="hidden" name="vpc_Amount" value="4995">
<input type="hidden" name="vpc_CardNum" value="5123456789012346">
<input type="hidden" name="vpc_cardExp" value="0405">
<!-- submit -->

<input type="submit" value="Pay Now">
</form>

```

## DR Fields for 2 Party Payments

### Required DR fields for 2-Party Payment Response

The DR contains the results of the DO that was processed by the Payment Server.

**Note :** *vpc\_TxnResponseCode* and *VPC\_Message* will always appear in the DR. Other fields may not be returned if the transaction was unsuccessful.

The fields that are included in a DR when using 2-Party Payments are:

Field Name	Required Optional Input	Field Type	Length	Example Value
<b>2-Party Payments - DR fields</b>				
vpc_Version	The value of the <i>vpc_Version</i> input field returned in the DR.			
	Input	Alphanumeric	1,2	1
vpc_Command	The value of the <i>vpc_Command</i> field returned in the DR.			
	Input	Alpha	1,16	pay
vpc_MerchTxnRef	The value of the <i>vpc_MerchTxnRef</i> field returned in the DR.			
	Input	Alphanumeric - Special characters	1,40	test1234/1
vpc_Merchant	The value of the <i>vpc_Merchant</i> input field returned in the DR.			
	Input	Alphanumeric - Special characters	1,16	TESTMERCHANT01
vpc_OrderInfo	The value of the <i>vpc_OrderInfo</i> input field returned in the DR.			
	Input	Alphanumeric - Special characters	1,34	test1234
vpc_Amount	The value of the <i>vpc_Amount</i> input field returned in the DR.			
	Input	Numeric	1,10	4995
vpc_Locale	Locale is not supplied in the DO but is returned in the DR. It is not used in 2-Party Payments.			
	Input	Alphanumeric	2,5	en_AU

Field Name	Required Optional Input	Field Type	Length	Example Value
vpc_TxnResponseCode	A response code that is generated by the Payment Server to indicate the status of the transaction. A <i>vpc_TxnResponseCode</i> of "0" (zero) indicates that the transaction was processed successfully and approved by the acquiring bank. Any other value indicates the transaction was declined. See Response Code.			
	Required	Alphanumeric	1	0
vpc_transactionNo	A unique number generated by the Payment Server. It is the reference value of the transaction in the Payment Server. This is the unique Payment Server OrderID transaction (Shopping Transaction) number that must be used for a Refund or Capture operation.			
	Required	Numeric	1,12	3465
vpc_Message	This is a message to indicate what sort of errors the transaction encountered.			
	Optional	Alphanumeric	10,200	Merchant [TESTCORE23] does not exist.
vpc_AcqResponseCode	Acquirer's Response Code is generated by the financial institution to indicate the status of the transaction. The results can vary between institutions so it is advisable to use the <i>vpc_TxnResponseCode</i> as it is consistent across all acquirers. It is only included for fault finding purposes.			
	Optional	Alphanumeric	2,3	00
vpc_ReceiptNo	This is also known as the Reference Retrieval Number (RRN), which is a unique identifier. This value is passed back to the cardholder for their records if the merchant application does not generate its own receipt number.			
	Optional	Alphanumeric - Special characters	1,12	RP12345
vpc_BatchNo	A date supplied by an acquirer to indicate when this transaction will be settled. If the batch has today's date then it will be settled the next day. When the acquirer closes the batch at the end of the day, the date will roll over to the next processing day's date.			
	Optional	Alphanumeric	1,8	20021021
vpc_AuthorizeId	A code issued by the acquiring bank to approve or deny the transaction. This may not always be supplied by all acquirers.			
	Optional	Alphanumeric	1,12	ABC12345
vpc_Card	A code issued by the Payment Server for the card type used by the cardholder for the transaction. For a list of codes, please refer to <b>Card Type Code</b> (see "Card Type Code" on page 91) on page 81.			
	Optional	Alphanumeric	2	ABC12345
Card Security Code (CSC) - DR fields				

Field Name	Required Optional Input	Field Type	Length	Example Value
vpc_AcqCSCRespCode	<p>The response code generated by the Issuing Bank in relation to the Card Security Code:</p> <p><b>For MasterCard Transactions:</b></p> <p><b>M</b> = Match  <b>N</b> = Not Match  <b>P</b> = Unable to Process  <b>U</b> = Issuer unregistered to process CVC 2/CVV2  <b>X</b> = Reserved for Magneprint</p> <p><b>For Visa Transactions</b></p> <p><b>M</b> = Match  <b>N</b> = Not Match  <b>P</b> = Unable to Process  <b>S</b> = CVV2 is on card but the merchant has indicated that CVV2 is not present  <b>U</b> = Issuer is not Visa-certified for CVV2, has not provided VISA encryption keys or both.</p> <p><b>For American Express:</b></p> <p>Currently, the CID is not supported for Amex transactions. However the Amex CID is indeed passed on for verification, the result is conclusive in the Transaction Response Code (vpc_TxnResponseCode).</p>			
	Optional	Alphanumeric	1	S
vpc_CSCResultCode	<p>The response code generated by the MiGS Payment Server in relation to the Card Security Code. However the field vpc_AcqCSCRespCode should be used for a more accurate representation of the CSC Response:</p> <p><b>M</b> = Match  <b>N</b> = Not Match  <b>P</b> = Unable to Process  <b>U</b> = Issuer unregistered to process CVC 2/CVV2</p>			
	Optional	Alphanumeric	1	M

**DR Fields -2 Party Style Pre-Authenticated Payment Transactions**

Field Name	Required Optional Input	Field Type	Length	Example Value
vpc_VerType		Echoed from request		

Field Name	Required Optional	Field Type	Length	Example Value
	Input			
	Required	Alphanumeric	3	3DS
vpc_VerStatus	A <b>status code</b> (see "Verified by Visa™ (Visa 3-Domain Secure) and MasterCard SecureCode™ (MasterCard 3-Domain Secure) Status Codes" on page 92) used by the Payment Server.			
	Required	Alphanumeric	28	Y
vpc_VerSecurityLevel	<p>The Verification Security Level is generated at the card issuer as a token to prove that the cardholder was enrolled and authenticated OK. It is shown for all transactions except those with authentication status "Failure". This field contains the security level to be used in the AUTH message.</p> <p>'05' - Fully Authenticated.</p> <p>'06' - Not authenticated, (cardholder not participating), liability shift.</p> <p>'07' - Not authenticated. Usually due to a system problem, for example the merchant password is invalid.</p>			
	Required	Alphanumeric	2	05
vpc_VerToken	Echoed from request			
	Optional	Alphanumeric		MDEyMzQ1Njc4OTAxMjM0NTY3OA
vpc_3DSXID	Echoed from request			
	Required	Alphanumeric		OTg3NjU0MzIxMDk4NzY1NDMyMTA
vpc_3DSECI	Echoed from request			
	Required	Alphanumeric	2	02
vpc_3DSStatus	Echoed from request			
	Required	Alphanumeric	1	Y
vpc_3DSenrolled	Echoed from request			
	Required	Alphanumeric	1	Y



## CHAPTER 10

## Additional Functionality API Reference Fields

In addition to the purchase and authorisation, there are a number of other functions that enable a merchant to process a range of payment options.

The table below summarises these functions.

Transaction	VPC Standard API	VPC Generic Request API		vpc_PaymentMethod*
	vpc_Command	vpc_RequestType	vpc_RequestCommand	
<b>Authorisation</b>	pay			CREDIT
<b>Capture</b>	capture			CREDIT
<b>Purchase</b>	pay			CREDIT DEBIT EBT
<b>Refund</b>	refund			CREDIT
<b>Void Capture</b>	voidCapture			CREDIT
<b>Void Refund</b>	voidRefund			CREDIT
<b>Void Purchase</b>	voidPurchase			CREDIT
<b>QueryDR</b>	queryDR			

### Capture

#### Digital Order Fields - Capture

The fields that can be included in a DO when using capture are:

Field Name	Required Optional	Field Type	Length	Example Value
vpc_Version		The version of the Virtual Payment Client API being used. The current version is 1.		
	Required	Alphanumeric	1,8	1
vpc_Command		Used to indicate the type of payment. The value, capture is used.		
	Required	Alpha	1,16	capture
		<p>A unique value created by the merchant to identify the DO. Used to track the progress of a transaction and allows it to be identified on the Payment Server should a communication's failure occur and the DR is not received. Can contain similar information to the <i>vpc_OrderInfo</i> field, but it must be unique.</p> <p>It may be in part an order number or invoice number, but it should also reflect the transaction attempt. For example, if a cardholder has insufficient funds on their card and you allow them to repeat the transaction with another credit card. The value may be test1234/1 on the first attempt, test1234/2 on the second attempt and test1234/3 on the third attempt. It can use text made up of any of the base US ASCII characters in the range, hexadecimal 20 to 126.</p>		

Field Name	Required Optional	Field Type	Length	Example Value
	Input			
	Required	Alphanumeric - Special characters	1,40	test1234/1
vpc_AccessCode	The access code authenticates a merchant on the Payment Server so that a merchant cannot access another merchant's MerchantId. The access code is provided to you when you registered your merchant profile with the Payment Provider.			
	Required	Alphanumeric	8	6ab89f3
vpc_Merchant	The unique merchant Id assigned to you by your Payment Provider.			
	Required	Alphanumeric	1,16	TESTMERCHANT01
vpc_transactionNo	The transaction reference number of the original authorisation or purchase.			
	Required	Numeric	1,12	123
vpc_Amount	The amount of the transaction in the smallest currency unit expressed as an integer. For example, if the transaction amount is \$49.95 then the amount in cents is 4995.			
	Required	Numeric	1,10	4995
vpc_User	This field is a special AMA user created to allow this function to operate.			
	Required	Alphanumeric	1,20	amauser
vpc_Password	The password used to authorise the AMA user to access this function.			
	Required	Alphanumeric	1,16	Password12

### Digital Receipt Fields - Capture

**Note :** *vpc\_TxnResponseCode* and *VPC\_Message* will always appear in the DR. Other fields may not be returned if the transaction was unsuccessful.

The fields included in a DR from the Virtual Payment Client when using captures are:

Field Name	Required Optional	Field Type	Length	Example Value
	Input			
vpc_Version	The value of the <i>vpc_Version</i> input field returned in the DR.			
	Input	Alphanumeric	1,8	1
vpc_Command	The value of the <i>vpc_Command</i> field returned in the DR.			
	Input	Alpha	1,16	capture
<i>vpc_MerchTxnRef</i>	The value of the <i>vpc_MerchTxnRef</i> field returned in the DR.			
	Input	Alphanumeric	1,40	test1234/1
vpc_Merchant	The value of the <i>vpc_Merchant</i> input field returned in the DR.			
	Input	Alphanumeric	1,16	TESTMERCHANT01
vpc_Amount	The value of the <i>vpc_Amount</i> input field returned in the DR.			
	Input	Numeric	1,10	4995
vpc_TxnResponseCode	A response code that is generated by the Payment Server to indicate the status of the transaction. A <i>vpc_TxnResponseCode</i> of "0" (zero) indicates that the transaction was processed successfully and approved by the acquiring bank. Any other value indicates the transaction was declined. See Response Code.			

Field Name	Required Optional	Field Type	Length	Example Value
	Input			
	Required	Alphanumeric	1	0
vpc_transactionNo	A unique number generated by the Payment Server and is the reference value of the transaction in the Payment Server. This value must be used for a Capture.			
	Required	Numeric	1,12	3465
vpc_Message	A message to indicate an error the transaction encountered.			
	Optional	Alphanumeric	10,200	Merchant [TESTCORE23] does not exist
vpc_AcqResponseCode	Acquirer's Response Code is generated by the bank to indicate the status of the transaction. The results can vary between institutions so it is advisable to use the vpc_TxnResponseCode as it is consistent across all acquirers. It is only included for fault finding purposes.			
	Optional	Alphanumeric	2,3	00
vpc_ReceiptNo	This is also known as the Reference Retrieval Number (RRN), a unique identifier. This value is passed back to the cardholder for their records if your application does not generate its own receipt number.			
	Optional	Alphanumeric	1,12	RP12345
vpc_BatchNo	A date supplied by an acquirer to indicate when this transaction will be settled. If the batch has today's date then it will be settled the next day. When the acquirer closes the batch at the end of the day, the date will roll over to the next processing day's date.			
	Optional	Alphanumeric	1,8	20021021
vpc_Authorized	A code issued by the acquiring bank to approve or deny the transaction. This may not always be supplied by all acquirers.			
	Optional	Alphanumeric	1,12	ABC12345
vpc_Card	A code issued by the Payment Server to detail the type of card the cardholder used for the transaction. For a list of codes, please see <b>Card Type Code</b> (see "Card Type Code" on page 91) on page 81.			
	Optional	Alphanumeric	2,4	ABC12345
vpc_ShopTransactionNo	The transaction reference number of the original authorisation or purchase transaction.			
	Optional	Numeric	1,21	3DS
vpc_AuthorisedAmount	The total amount of the original authorisation transaction.			
	Optional	Numeric	1,10	4995
vpc_CapturedAmount	The amount of the capture transaction in the smallest currency unit expressed as an integer. For example, if the transaction amount is \$49.95 then the amount in cents is 4995.			
	Optional	Numeric	1,10	4995
vpc_TicketNo	Allows you to include a ticket number, such as an airline ticket number in the DO. The ticket number is stored on the Payment Server database for that transaction and returned in the DR for capture transactions.			
	Optional	Alphanumeric	1,15	Any data

## Refund

### Digital Order Fields - Refund

The fields that can be included in a DO to the Virtual Payment Client when using refund are:

Field Name	Required Optional Input	Field Type	Length	Example Value
vpc_Version	The version of the Virtual Payment Client API being used. The current version is 1.			
	Required	Alphanumeric	1,8	1
vpc_Command	Used to indicate the type of payment. For refunds, the value 'refund' is used.			
	Required	Alpha	1,16	refund
vpc_MerchTxnRef	A unique value created by the merchant to identify the DO. It is used to track the progress of a transaction and allows it to be identified on the Payment Server should a communication's failure occur and the DR is not received. It can contain similar information to the <i>vpc_OrderInfo</i> field, but it must be unique. It may be in part an order number or invoice number, but it should also reflect the transaction attempt. For example, if a cardholder has insufficient funds on their card and you allow them to repeat the transaction with another credit card. The value may be test1234/1 on the first attempt, test1234/2 on the second attempt and test1234/3 on the third attempt. It can use text made up of any of the base US ASCII characters in the range, hexadecimal 20 to 126.			
	Required	Alphanumeric - Special characters	1,40	test1234/1
vpc_AccessCode	The access code authenticates a merchant on the Payment Server so that a merchant cannot access another merchant's MerchantId. The access code is provided to you when you registered your merchant profile with the Payment Provider.			
	Required	Alphanumeric	8	6ab89f3
vpc_Merchant	The unique merchant Id assigned to you by your Payment Provider.			
	Required	Alphanumeric	1,16	TESTMERCHANT01
vpc_transactionNo	The transaction reference number of the original Authorisation or Purchase transaction.			
	Required	Numeric	1,12	123
vpc_Amount	The amount of the refund transaction in the smallest currency unit expressed as an integer. For example, if the transaction amount is \$49.95 then the amount in cents is 4995.			
	Required	Numeric	1,10	4995
vpc_User	This field is a special AMA user created to allow this function to operate.			
	Required	Alphanumeric	1,20	amauser
vpc_Password	The password used to authorise the AMA user access to this function.			
	Required	Alphanumeric	1,16	password12

**Digital Request Fields - Refund**

**Note :** *vpc\_TxnResponseCode* and *VPC\_Message* will always appear in the DR. Other fields may not be returned if the transaction was unsuccessful.

The fields included in a DR from the Virtual Payment Client when using refunds are:

Field Name	Required Optional	Field Type	Length	Example Value
vpc_Version		The value of the <i>vpc_Version</i> input field returned in the DR.		
	Input	Alphanumeric	1,8	1
vpc_Command		The value of the <i>vpc_Command</i> field returned in the DR.		
	Input	Alpha	1,16	refund
vpc_MerchTxnRef		The value of the <i>vpc_MerchTxnRef</i> field returned in the DR.		
	Input	Alphanumeric - Special characters	1,40	test1234/1
vpc_Merchant		The value of the <i>vpc_Merchant</i> input field returned in the DR.		
	Input	Alphanumeric - Special characters	1,16	TESTMERCHANT01
vpc_Amount		The value of the <i>vpc_Amount</i> input field returned in the DR.		
	Input	Numeric	1,10	4995
vpc_TxnResponseCode		A response code that is generated by the Payment Server to indicate the status of the transaction. A <i>vpc_TxnResponseCode</i> of "0" (zero) indicates that the transaction was processed successfully and approved by the acquiring bank. Any other value indicates the transaction was declined. See Response Code.		
	Required	Alphanumeric	1	0
vpc_transactionNo		A unique number generated by the Payment Server. It is the reference value of the transaction in the Payment Server. This is the value that must be used for a Refund.		
	Required	Numeric	1,12	3465
vpc_Message		A message to indicate any errors the transaction may have encountered.		
	Optional	Alphanumeric	10,200	Merchant [TESTCORE23] does not exist
vpc_AcqResponseCode		Acquirer's Response Code is generated by the financial institution to indicate the status of the transaction. The results can vary between institutions so it is advisable to use the <i>vpc_TxnResponseCode</i> as it is consistent across all acquirers. It is only included for fault finding purposes.		
	Optional	Alphanumeric	2,3	00
vpc_ReceiptNo		This is also known as the Reference Retrieval Number (RRN), which is a unique identifier. This value is passed back to the cardholder for their records if the merchant application does not generate its own receipt number.		

Field Name	Required Optional	Field Type	Length	Example Value
	Input			
	Optional	Alphanumeric	1,12	RP12345
vpc_BatchNo	A date supplied by an acquirer to indicate when this transaction will be settled. If the batch has today's date then it will be settled the next day. When the acquirer closes the batch at the end of the day, the date will roll over to the next processing day's date.			
	Optional	Alphanumeric	1,8	20021021
vpc_AuthorizedId	A code issued by the acquiring bank to approve or deny the transaction. This may not always be supplied by all acquirers.			
	Optional	Alphanumeric	1,12	ABC12345
vpc_Card	A code issued by the Payment Server to detail the type of card the cardholder used for this transaction. For a list of codes, please see <b>Card Type Code</b> (see "Card Type Code" on page 91) on page 81.			
	Optional	Alphanumeric	2,4	ABC12345
vpc_ShopTransactionNo	The transaction reference number of the original authorisation or purchase transaction.			
	Optional	Numeric	1,19	4567
vpc_AuthorisedAmount	The total amount of the original authorisation transaction.			
	Optional	Numeric	1,10	4995
vpc_RefundedAmount	The amount of the refund transaction in the smallest currency unit expressed as an integer. For example, if the transaction amount is \$49.95 then the amount in cents is 4995.			
	Optional	Numeric	1,10	4995
vpc_TicketNo	Allows you to include a ticket number, such as an airline ticket number in the DO. The ticket number is stored on the Payment Server database for that transaction. The ticket number is stored on the Payment Server database for that transaction and returned in the DR for refunds.			
	Optional	Alphanumeric - Special characters	1,15	Any data

## Void Capture

### Digital Order Fields-Void Capture

The fields that can be included in a DO to the Virtual Payment Client when using void capture are:

Field Name	Required Optional	Field Type	Length	Example Value
	Input			
vpc_Version	The version of the Virtual Payment Client API being used. The current version is 1.			
	Required	Alphanumeric	1,8	1
vpc_Command	Used to indicate the type of transaction. For refunds, the value 'voidCapture' is used.			

Field Name	Required Optional	Field Type	Length	Example Value
	Input			
	Required	Alpha	1,16	voidCapture
<i>vpc_MerchTxnRef</i>	<p>A unique value created by you to identify the DO. It is used to track the progress of a transaction and allows it to be identified on the Payment Server should a communication's failure occur and the DR is not received.</p> <p>It can contain similar information to the <i>vpc_OrderInfo</i> field, but it must be unique.</p> <p>It may be in part an order number or invoice number, but it should also reflect the transaction attempt. For example, if a cardholder has insufficient funds on their card and you allow them to repeat the transaction with another credit card. The value may be test1234/1 on the first attempt, test1234/2 on the second attempt and test1234/3 on the third attempt.</p> <p>It can use text made up of any of the base US ASCII characters in the range, hexadecimal 20 to 126.</p>			
	Required	Alphanumeric - Special characters	1,40	test1234/1
<i>vpc_AccessCode</i>	<p>The access code authenticates a merchant on the Payment Server so that a merchant cannot access another merchant's MerchantId. The access code is provided to you when you registered your merchant profile with the Payment Provider.</p>			
	Required	Alphanumeric	8	6ab89f3
<i>vpc_Merchant</i>	<p>The unique merchant Id assigned to you by your Payment Provider.</p>			
	Required	Alphanumeric	1,16	TESTMERCHANT01
<i>vpc_transactionNo</i>	<p>The transaction reference number of the original Authorisation transaction.</p>			
	Required	Numeric	1,12	123
<i>vpc_User</i>	<p>This field is a special AMA user created to allow this function to operate.</p>			
	Required	Alphanumeric	1,20	amauser
<i>vpc_Password</i>	<p>The password used to authorise the AMA user access to this function.</p>			
	Required	Alphanumeric	1,16	password12

### Digital Receipt Fields - Void Capture

**Note :** *vpc\_TxnResponseCode* and *VPC\_Message* will always appear in the DR. Other fields may not be returned if the transaction was unsuccessful.

The fields included in a DR from the Virtual Payment Client when using void captures are:

Field Name	Required Optional	Field Type	Length	Example Value
	Input			
<i>vpc_Version</i>	<p>The value of the <i>vpc_Version</i> input field returned in the DR.</p>			
	Input	Alphanumeric	1,8	1
<i>vpc_Command</i>	<p>The value of the <i>vpc_Command</i> field returned in the DR.</p>			

Field Name	Required Optional	Field Type	Length	Example Value
	Input			
	Input	Alpha	1,16	voidCapture
<i>vpc_MerchTxnRef</i>	The value of the <i>vpc_MerchTxnRef</i> field returned in the DR.			
	Input	Alphanumeric - Special characters	1,40	test1234/1
<i>vpc_Merchant</i>	The value of the <i>vpc_Merchant</i> input field returned in the DR.			
	Input	Alphanumeric - Special characters	1,16	TESTMERCHANT01
<i>vpc_TxnResponseCode</i>	A response code that is generated by the Payment Server to indicate the status of the transaction. A <i>vpc_TxnResponseCode</i> of "0" (zero) indicates that the transaction was processed successfully and approved by the acquiring bank. Any other value indicates the transaction was declined. See Response Codes.			
	Required	Alphanumeric	1	0
<i>vpc_transactionNo</i>	A unique number generated by the Payment Server. It is the reference value of the transaction in the Payment Server. This is the value that must be used for a Refund.			
	Required	Numeric	1,12	3465
<i>vpc_Message</i>	A message to indicate any errors the transaction may have encountered.			
	Optional	Alphanumeric	10,200	Merchant [TESTCORE23] does not exist
<i>vpc_AcqResponseCode</i>	Acquirer's Response Code is generated by the financial institution to indicate the status of the transaction. The results can vary between institutions so it is advisable to use the <i>vpc_TxnResponseCode</i> as it is consistent across all acquirers. It is only included for fault finding purposes.			
	Optional	Alphanumeric	2,3	00
<i>vpc_ReceiptNo</i>	This is also known as the Reference Retrieval Number (RRN), which is a unique identifier. This value is passed back to the cardholder for their records if the merchant application does not generate its own receipt number.			
	Optional	Alphanumeric	1,12	RP12345
<i>vpc_BatchNo</i>	A date supplied by an acquirer to indicate when this transaction will be settled. If the batch has today's date then it will be settled the next day. When the acquirer closes the batch at the end of the day, the date will roll over to the next processing day's date.			
	Optional	Alphanumeric	1,8	20021021
<i>vpc_Authorizeld</i>	A code issued by the acquiring bank to approve or deny the transaction. This may not always be supplied by all acquirers.			
	Optional	Alphanumeric	1,12	ABC12345
<i>vpc_Card</i>	A code issued by the Payment Server to detail the type of card the cardholder used for this transaction. For a list of codes, please see <b>Card Type Code</b> (see "Card Type Code" on page 91) on page 81.			
	Optional	Alphanumeric	2,4	ABC12345
<i>vpc_ShopTransactionNo</i>	The transaction reference number of the original authorisation transaction.			

Field Name	Required Optional	Field Type	Length	Example Value
	Input			
	Optional	Numeric	1,19	4567
vpc_AuthorisedAmount	The total amount of the original authorisation transaction.			
	Optional	Numeric	1,10	5999
vpc_RefundedAmount	The net refunded amount in the smallest currency unit expressed as an integer. For example, if the transaction amount is \$49.95 then the amount in cents is 4995.			
	Optional	Numeric	1,10	4995
vpc_TicketNo	Allows you to include a ticket number, such as an airline ticket number in the DO. The ticket number is stored on the Payment Server database for that transaction. The ticket number is stored on the Payment Server database for that transaction and returned in the DR for refunds.			
	Optional	Alphanumeric - Special characters	1,15	Any data
vpc_CapturedAmount	The net captured amount in the smallest currency unit expressed as an integer. For example, if the transaction amount is \$49.95 then the amount in cents is 4995.			
	Optional	Numeric	1,10	4995

## Void Refund

### Digital Order Fields-Void Refund

The fields that can be included in a DO to the Virtual Payment Client when using void refund are:

Field Name	Required Optional	Field Type	Length	Example Value
	Input			
vpc_Version	The version of the Virtual Payment Client API being used. The current version is 1.			
	Required	Alphanumeric	1,8	1
vpc_Command	Used to indicate the type of payment. For void refunds, the value 'voidRefund' is used.			

Field Name	Required Optional	Field Type	Length	Example Value
	Input			
	Required	Alpha	1,16	voidRefund
<i>vpc_MerchTxnRef</i>	<p>A unique value created by you to identify the DO. It is used to track the progress of a transaction and allows it to be identified on the Payment Server should a communication's failure occur and the DR is not received.</p> <p>It can contain similar information to the <i>vpc_OrderInfo</i> field, but it must be unique.</p> <p>It may be in part an order number or invoice number, but it should also reflect the transaction attempt. For example, if a cardholder has insufficient funds on their card and you allow them to repeat the transaction with another credit card. The value may be test1234/1 on the first attempt, test1234/2 on the second attempt and test1234/3 on the third attempt.</p> <p>It can use text made up of any of the base US ASCII characters in the range, hexadecimal 20 to 126.</p>			
	Required	Alphanumeric - Special characters	1,40	test1234/1
<i>vpc_AccessCode</i>	<p>The access code authenticates a merchant on the Payment Server so that a merchant cannot access another merchant's MerchantId. The access code is provided to you when you registered your merchant profile with the Payment Provider.</p>			
	Required	Alphanumeric	8	6ab89f3
<i>vpc_Merchant</i>	<p>The unique merchant Id assigned to you by your Payment Provider.</p>			
	Required	Alphanumeric	1,16	TESTMERCHANT01
<i>vpc_transactionNo</i>	<p>The transaction reference number of the original Authorisation or Purchase transaction.</p>			
	Required	Numeric	1,12	123
<i>vpc_User</i>	<p>This field is a special AMA user created to allow this function to operate.</p>			
	Required	Alphanumeric	1,20	amauser
<i>vpc_Password</i>	<p>The password used to authorise the AMA user access to this function.</p>			
	Required	Alphanumeric	1,16	password12

### Digital Receipt Fields - Void Refund

The fields included in a DR from the Virtual Payment Client when using voids are:

Field Name	Required Optional	Field Type	Length	Example Value
<i>vpc_Version</i>	Input	Alphanumeric	1,8	1
<i>vpc_Command</i>	Input	Alpha	1,16	voidRefund
<i>vpc_MerchTxnRef</i>	<p>The value of the <i>vpc_MerchTxnRef</i> field returned in the DR.</p>			

Field Name	Required Optional	Field Type	Length	Example Value
	Input	Alphanumeric - Special characters	1,40	test1234/1
vpc_Merchant	The value of the <i>vpc_Merchant</i> input field returned in the DR.			
	Input	Alphanumeric - Special characters	1,16	TESTMERCHANT01
vpc_TxnResponseCode	A response code that is generated by the Payment Server to indicate the status of the transaction. A <i>vpc_TxnResponseCode</i> of "0" (zero) indicates that the transaction was processed successfully and approved by the acquiring bank. Any other value indicates the transaction was declined. See Response Code.			
	Required	Alphanumeric	1	0
vpc_transactionNo	A unique number generated by the Payment Server. It is the reference value of the transaction in the Payment Server. This is the value that must be used for a Refund.			
	Required	Numeric	1,12	3465
vpc_Message	A message to indicate any errors the transaction may have encountered.			
	Optional	Alphanumeric	10,200	Merchant [TESTCORE23] does not exist
vpc_AcqResponseCode	Acquirer's Response Code is generated by the financial institution to indicate the status of the transaction. The results can vary between institutions so it is advisable to use the <i>vpc_TxnResponseCode</i> as it is consistent across all acquirers. It is only included for fault finding purposes.			
	Optional	Alphanumeric	2,3	00
vpc_ReceiptNo	This is also known as the Reference Retrieval Number (RRN), which is a unique identifier. This value is passed back to the cardholder for their records if the merchant application does not generate its own receipt number.			
	Optional	Alphanumeric	1,12	RP12345
vpc_BatchNo	A date supplied by an acquirer to indicate when this transaction will be settled. If the batch has today's date then it will be settled the next day. When the acquirer closes the batch at the end of the day, the date will roll over to the next processing day's date.			
	Optional	Alphanumeric	1,8	20021021
vpc_Authorizeld	A code issued by the acquiring bank to approve or deny the transaction. This may not always be supplied by all acquirers.			
	Optional	Alphanumeric	1,12	ABC12345
vpc_Card	A code issued by the Payment Server to detail the type of card the cardholder used for this transaction. For a list of codes, please see <b>Card Type Code</b> (see "Card Type Code" on page 91) on page 81.			
	Optional	Alphanumeric	2,4	ABC12345
vpc_ShopTransactionNo	The transaction reference number of the original authorisation or purchase transaction.			
	Optional	Numeric	1,19	3DS
vpc_AuthorisedAmount	The total amount of the original authorisation or purchase transaction.			

Field Name	Required Optional	Field Type	Length	Example Value
	Input			
	Optional	Numeric	1,10	4995
vpc_RefundedAmount	The net refunded in the smallest currency unit expressed as an integer. For example, if the transaction amount is \$49.95 then the amount in cents is 4995.			
	Optional	Numeric	1,10	4995
vpc_TicketNo	Allows you to include a ticket number, such as an airline ticket number in the DO. The ticket number is stored on the Payment Server database for that transaction. The ticket number is stored on the Payment Server database for that transaction and returned in the DR for refunds.			
	Optional	Alphanumeric - Special characters	1,15	Any data
vpc_CapturedAmount	The net captured amount in the smallest currency unit expressed as an integer. For example, if the transaction amount is \$49.95 then the amount in cents is 4995.			
	Optional	Numeric	1,10	4995

## Void Purchase

### Digital Order Fields-Void Purchase

The fields that can be included in a DO to the Virtual Payment Client when using void capture are:

Field Name	Required Optional	Field Type	Length	Example Value
	Input			
vpc_Version	The version of the Virtual Payment Client API being used. The current version is 1.			
	Required	Alphanumeric	1,8	1
vpc_Command	Used to indicate the type of transaction. For void purchases, the value 'voidPurchase' is used.			
	Required	Alpha	1,16	voidPurchase
vpc_MerchTxnRef	<p>A unique value created by you to identify the DO. It is used to track the progress of a transaction and allows it to be identified on the Payment Server should a communication's failure occur and the DR is not received.</p> <p>It can contain similar information to the <i>vpc_OrderInfo</i> field, but it must be unique.</p> <p>It may be in part an order number or invoice number, but it should also reflect the transaction attempt. For example, if a cardholder has insufficient funds on their card and you allow them to repeat the transaction with another credit card. The value may be test1234/1 on the first attempt, test1234/2 on the second attempt and test1234/3 on the third attempt.</p> <p>It can use text made up of any of the base US ASCII characters in the range, hexadecimal 20 to 126.</p>			

Field Name	Required Optional	Field Type	Length	Example Value
	Input			
	Required	Alphanumeric - Special characters	1,40	test1234/1
vpc_AccessCode	The access code authenticates a merchant on the Payment Server so that a merchant cannot access another merchant's MerchantId. The access code is provided when you registered your merchant profile with the Payment Provider.			
	Required	Alphanumeric	8	6ab89f3
vpc_Merchant	The unique merchant Id assigned to you by your Payment Provider.			
	Required	Alphanumeric	1,16	TESTMERCHANT01
vpc_transactionNo	The transaction reference number of the original Purchase transaction.			
	Required	Numeric	1,12	123
vpc_User	This field is a special AMA user created to allow this function to operate.			
	Required	Alphanumeric	1,20	amauser
vpc_Password	The password used to authorise the AMA user access to this function.			
	Required	Alphanumeric	1,16	password12

#### Digital Receipt Fields - Void Purchase

**Note :** *vpc\_TxnResponseCode* and *VPC\_Message* will always appear in the DR. Other fields may not be returned if the transaction was unsuccessful.

The fields included in a DR from the Virtual Payment Client when using voids are:

Field Name	Required Optional	Field Type	Length	Example Value
	Input			
vpc_Version	The value of the <i>vpc_Version</i> input field returned in the DR.			
	Input	Alphanumeric	1,8	1
vpc_Command	The value of the <i>vpc_Command</i> field returned in the DR.			
	Input	Alpha	1,16	voidPurchase
vpc_MerchTxnRef	The value of the <i>vpc_MerchTxnRef</i> field returned in the DR.			
	Input	Alphanumeric - Special characters	1,40	test1234/1
vpc_Merchant	The value of the <i>vpc_Merchant</i> input field returned in the DR.			
	Input	Alphanumeric - Special characters	1,16	TESTMERCHANT01
vpc_TxnResponseCode	A response code that is generated by the Payment Server to indicate the status of the transaction. A <i>vpc_TxnResponseCode</i> of "0" (zero) indicates that the transaction was processed successfully and approved by the acquiring bank. Any other value indicates the transaction was declined. See Response Code.			

Field Name	Required Optional Input	Field Type	Length	Example Value
	Required	Alphanumeric	1	0
vpc_transactionNo	A unique number generated by the Payment Server. It is the reference value of the transaction in the Payment Server. This is the value that must be used for a Refund.			
	Required	Numeric	1,12	3465
vpc_Message	A message to indicate any errors the transaction may have encountered.			
	Optional	Alphanumeric	10,200	Merchant [TESTCORE23] does not exist
vpc_AcqResponseCode	Acquirer's Response Code is generated by the financial institution to indicate the status of the transaction. The results can vary between institutions so it is advisable to use the vpc_TxnResponseCode as it is consistent across all acquirers. It is only included for fault finding purposes.			
	Optional	Alphanumeric	2,3	00
vpc_ReceiptNo	This is also known as the Reference Retrieval Number (RRN), which is a unique identifier. This value is passed back to the cardholder for their records if the merchant application does not generate its own receipt number.			
	Optional	Alphanumeric	1,12	RP12345
vpc_BatchNo	A date supplied by an acquirer to indicate when this transaction will be settled. If the batch has today's date then it will be settled the next day. When the acquirer closes the batch at the end of the day, the date will roll over to the next processing day's date.			
	Optional	Alphanumeric	1,8	20021021
vpc_Authorizeld	A code issued by the acquiring bank to approve or deny the transaction. This may not always be supplied by all acquirers.			
	Optional	Alphanumeric	1,12	ABC12345
vpc_Card	A code issued by the Payment Server to detail the type of card the cardholder used for this transaction. For a list of codes, please see <b>Card Type Code</b> (see "Card Type Code" on page 91) on page 81.			
	Optional	Alphanumeric	2,4	ABC12345
vpc_ShopTransactionNo	The transaction reference number of the original purchase transaction.			
	Optional	Numeric	1,19	1234
vpc_AuthorisedAmount	The net amount of the original purchase transaction. If successful, this should be 0.			
	Optional	Numeric	1,10	4995
vpc_RefundedAmount	The net refunded amount in the smallest currency unit expressed as an integer. For example, if the transaction amount is \$49.95 then the amount in cents is 4995.			
	Optional	Numeric	1,10	4995
vpc_TicketNo	Use to include a ticket number, such as an airline ticket number in the DO. The ticket number is stored on the Payment Server database for that transaction. The ticket number is stored on the Payment Server database for that transaction and returned in the DR for refunds.			

Field Name	Required Optional Input	Field Type	Length	Example Value
vpc_CapturedAmount	Optional	Alphanumeric - Special characters	1,15	Any data
	The net captured amount. In a successful void purchase, this should be 0			
	Optional	Numeric	1,10	0

## Bypass Card Selection Page on the Payment Server

This is used in 3-Party Payments to bypass the Payment Server payments page that displays the logos of all the cards the payment processor will accept.

### Note:

To use this option, the merchant must have the appropriate privileges set in the Payment Server or the transaction will fail.

### DO Fields - Bypass Card Selection Page

The fields that are included in a DO when using Bypass Card Selection are:

Field Name	Required Optional Input	Field Type	Length	Example Value
vpc_Gateway	This field determines the VPC gateway that will be used. The field is case sensitive, and must comply with the gateways that are valid in the Payment Server. The value used here will always be <i>ssl</i> .			
	Optional	Alpha	3	ssl
vpc_Card	A code issued by the Payment Server for the card type used by the cardholder in the transaction. For a list of card type codes, please refer to page 81.			
	Optional	Alphanumeric	2,4	MC

### DR Fields - Bypass Card Selection Page

The Bypass Card Selection page functionality does not return any extra fields in the DR.

## QueryDR Transaction

The QueryDR command allows you to search for a lost DR. The search is performed on the primary key - **MerchTxnRef**, which is why the *vpc\_MerchTxnRef* field needs to be a unique value.

If there are duplicate *vpc\_MerchTxnRef* numbers, while the query returns the most recent transaction encrypted DR, a flag indicates the presence of multiple transactions that meet the criteria. If the query result returned is not the correct one, use Merchant Administration on the Payment Server to search for the correct transaction.

**DO Fields - QueryDR**

The fields that are included in a DO when using QueryDR are:

Field Name	Required Optional	Field Type	Length	Example Value
vpc_Version	The version of the API being used. The current version is 1.			
	Required	Alphanumeric	1,8	1
vpc_Command	This indicates the type of transaction.			
	Required	Alpha	1,16	queryDR
vpc_AccessCode	The access code authenticates a merchant on the Payment Server so that a merchant cannot access another merchant's MerchantId. The access code is provided to you when you registered your merchant profile with the Payment Provider.			
	Required	Alphanumeric	8	6ab89f3
vpc_Merchant	The unique merchant Id assigned to you by your Payment Provider.			
	Required	Alphanumeric	1,16	TESTMERCH ANT01
vpc_MerchTxnRef	It is the primary key used to search the progress of a transaction in the event of a communication's failure where no DR is received.			
	Required	Alphanumeric - Special characters	1,40	test1234/1
vpc_User	This field is a special AMA user created to use this function.			
	Required	Alphanumeric	1,20	amauser
vpc_Password	The password used to authorise the AMA user access to this function.			
	Required	Alphanumeric	1,16	password12

**DR Details - QueryDR**

**Note :** *vpc\_TxnResponseCode* and *VPC\_Message* will always appear in the DR. Other fields may not be returned if the transaction was unsuccessful.

The fields returned in the DR are the same as the original transaction, but includes two additional DR fields. The fields that are included in a DR when using QueryDR are:

Field Name	Required Optional	Field Type	Length	Example Value
vpc_DRExists	This key is used to determine if the QueryDR command returned any search results. If the value is "Y", then there is at least one <i>vpc_MerchTxnRef</i> number result matching the search criteria.			
	Optional	Alpha	1	Y
vpc_FoundMultipleDRs	This is used after the previous command to determine if there are multiple results. If the value is "Y", then there are multiple <i>MerchTxnRef</i> numbers matching the search criteria.			
	Optional	Alpha	1	N

## CHAPTER 11

# Troubleshooting and FAQs

## In This Chapter

Troubleshooting .....	83
Frequently Asked Questions .....	86

---

## Troubleshooting

This section contains suggestions and solutions to problems that may occur with your integration.

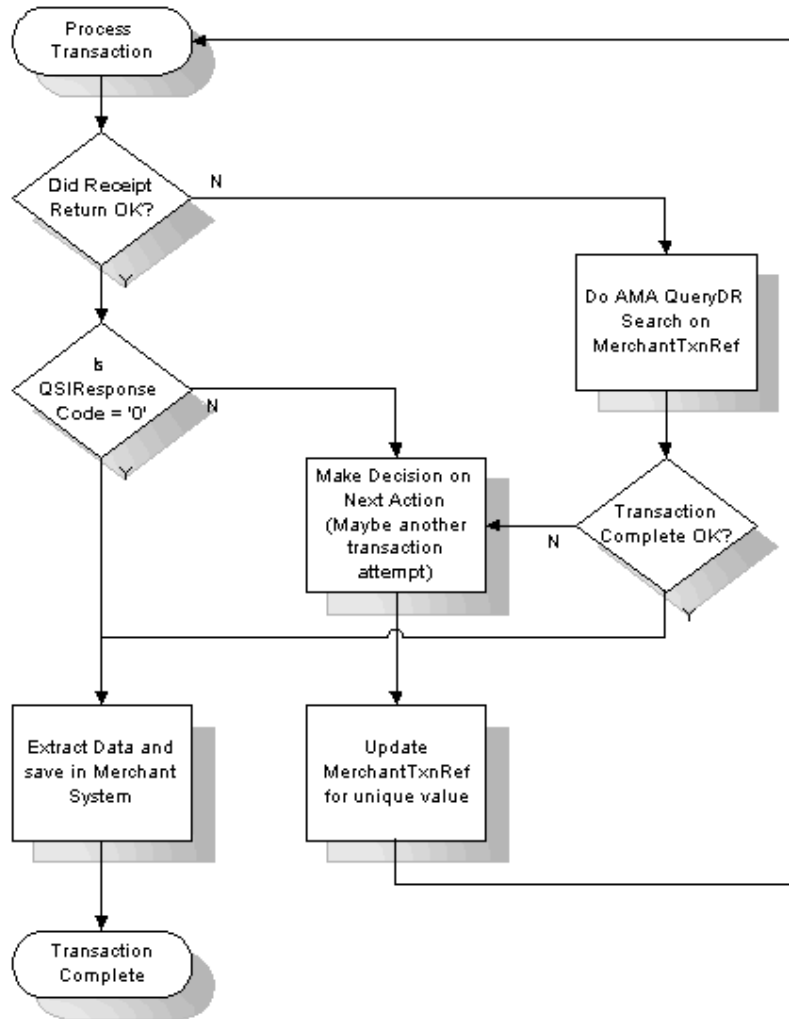
### What happens if a Transaction Response fails to come back?

To deal with a Transaction Response that fails to come back:

- 1 Flag the transaction as having an error, so that it needs to be manually checked using Merchant Administration on the Payment Server.

Or,

Use the Advanced Merchant Administration (AMA), *QueryDR* command to search the Payment Server database for the transaction. The *vpc\_MerchTxnRef* is used as the transaction identifier when searching using *QueryDR* command.



Since the Transaction Response has failed to come back, there is no transaction number available from the Payment Server to identify the transaction in question, and this is why you use the *vpc\_MerchTxnRef*. It is important to have a unique *vpc\_MerchTxnRef* for every transaction otherwise the query could return multiple results. Only the most recent transaction is returned in the *QueryDR* command if there are multiple results, but this may not be the transaction you are concerned with.

- If the *vpc\_TxnResponseCode* was successful
 

When you find the required *vpc\_MerchTxnRef* in the *QueryDR*, check the *vpc\_TxnResponseCode* field to see if it is successful (should be equal to '0'). If the *vpc\_TxnResponseCode* is 0, then the transaction is successful and you just need to extract the relevant data details from the *QueryDR* results for your records.
- If the *vpc\_TxnResponseCode* code was not successful
 

If the *vpc\_TxnResponseCode* is not 0, you need to determine the next course of action based on what you would do if the *vpc\_TxnResponseCode* were not 0 in a normal Transaction Response coming back from the Payment Server.
- If you did not find a *QueryDR* result
 

If you query the Payment Server for the *vpc\_MerchTxnRef* using the *QueryDR* call and you do not receive any results, i.e. *vpc\_DRExists* = 'N', then it is safe to repeat the transaction. It is safe to use the same *vpc\_MerchTxnRef*, as the existing one does not show up in the Payment Server's database and therefore it was never processed.

- If you find multiple *QueryDR* results

If the *QueryDR* is flagged as having multiple results (returns 'Y' in the *vpc\_FoundMultipleDRs* field), then the *vpc\_MerchTxnRef* is not unique. This is the primary reason for implementing a unique *vpc\_MerchTxnRef* for every transaction. *QueryDR* returns the latest transaction but this may not be the one you are after.

This solution requires more data capture and processing, but it is only necessary when you don't have a unique *vpc\_MerchTxnRef* number.

## What do we do if a Session Timeout occurs?

It is possible that while a cardholder is entering their card details at the Payment Server, the session is broken (say a communication failure due to a modem connection dropping off). If this occurs, a cardholder will lose their session. Even if they come back to your site, they will have a new session, and their old session will never be completed.

To determine the status of the lost transaction, you will need to perform a **QueryDR transaction** (see "QueryDR Transaction" on page 81) based on the original *vpc\_MerchTxnRef*.

## What does a Payment Authentication Status of "A" mean?

An authentication state of "A" indicates that the authentication transaction failed when the Payment Server tried to authenticate itself with the Directory Server.

**Note:** The unauthenticated payment transaction will proceed, however no liability shift to the card issuer occurs, that is, the merchant is liable for any possible fraud.

A possible reason for the failure is that the 3-D Secure (for Verified by Visa™ (Visa 3-Domain Secure) and MasterCard SecureCode™ (MasterCard 3-Domain Secure)) merchant ID or password is set incorrectly for a merchant profile in the Payment Server Merchant Manager, that is, the merchant's Verified by Visa username and password (for Visa) or SecureCode username and password (for MasterCard) have not been set correctly by the Merchant Service Organisation (MSO) in the merchant profile.

## Does the Cardholder's Internet browser need to support cookies?

The Virtual Payment Client interface requires a cardholder's browser to support cookies for all 3-Party Payments.

## How do I know if a transaction has been approved?

All approved transactions are represented with a **vpc\_TxnResponseCode** of '0' (zero) from the Payment Server. Any other code represents a declined or failed transaction.

## What is an outage?

An outage is considered a "production fault" as it means that the Payment Server is temporarily offline; for example for maintenance and upgrades, etc.

During an outage, all transactions are declined with an error message indicating that the service is currently unavailable.

---

## Frequently Asked Questions

### Is a Shopping Cart required?

It is not necessary to have a shopping cart. All that is required is that the transaction information is within the Transaction Request passed to the Payment Server.

### What is Merchant Administration?

Merchant Administration allows merchants to use an Internet browser to monitor and manage electronic transactions through a series of easy to use pages. It allows merchants to interactively perform historical searches, captures, refunds and setup activities.

To use Merchant Administration, you need access to the Internet through a browser (such as Internet Explorer or Netscape) , your Payment Providers URL (or web site address) and a merchant profile. The merchant profile is a record of your details and privileges, which are stored on the Payment Server. For more details, please refer to the Merchant Administration User Guide.

### How much will it cost to keep the payment site running?

The Virtual Payment Client is very stable and is not difficult to keep running and requires no more maintenance than the web server itself.

### Does the Payment Server handle large peaks in transaction volumes?

The Payment Server queues pending transactions so transactions are not lost, although the cardholder may at times notice a slight delay when transactional loads are extremely high.

### How long will an authorisation be valid on a cardholder account?

This depends on the Financial Institution who issued the card to the cardholder. Each card Issuer defines the authorisation expiry period in which they hold the funds on the cardholder's account, while they wait for the arrival of the capture transaction. Generally it is 5-8 processing days, before the authorisation purges from the cardholder account and access to the funds are released back to the cardholder.

## What is the RRN and how do I use it?

RRN (Reference Retrieval Number) is a unique number for a particular MerchantId. This is the value that is passed back to the cardholder for their records. You cannot search for this field in Merchant Administration, but it is displayed in Merchant Administration on the transaction details pages as the Reference Retrieval Number (RRN). It is one of the fields returned in a *QueryDR* and the transaction result (captures, refunds).

The RRN is useful when your application does not provide a receipt number. The RRN can be viewed in Merchant Administration.

## What is the difference between RRN, MerchTxnRef, OrderInfo, AuthorizeId and TransactionNo ?

- *RRN* (Reference Retrieval Number) is a unique number assigned to each transaction for a particular MerchantId. This is the value that is passed back to the cardholder for their records. You cannot search on this field in Merchant Administration, but it is displayed in Merchant Administration on the transaction details pages as the Reference Retrieval Number (RRN). It is one of the fields returned in a *queryDR* and the transaction result (captures, refunds).
- *MerchTxnRef* is generated by your merchant application. Ideally it should be a unique value for each transaction and you should retain this number so that transactions can be searched for in your application and the Payment Server. See Merchant Transaction Reference (**vpc\_MerchTxnRef**)
- *OrderInfo* is also generated by your application. It should also be a unique value for each order, which you should retain so that you can search for the transaction in your application and the Payment Server.
- *AuthorizeId* is an identifier from the Acquiring Bank, which is in the Transaction Response for the authorisation. This field cannot be searched for in Merchant Administration, but it is displayed in Merchant Administration as the Authorisation Code. It is one of the fields returned in a transaction result and an *AMA QueryDR*.
- *TransactionNo or TransactionNo* is a unique number for each MerchantId generated by the Payment Server that is called the OrderID or shopping transaction number. The OrderID is the key reference value for transactions when using *AMA* transactional functions like captures and refunds.



## CHAPTER 12

**References - Virtual Payment Client**

## Returned Response Codes

The *vpc\_TxnResponseCode* is a response code generated by the Payment Server that shows whether the transaction was successful or not. This response code can also be used to detect an error.

Any response code other than '0' is a declined/failed transaction. If the transaction is an error condition it can be determined by executing the `DigitalReceipt.ERROR` command. If this returns a response it is an error condition, not only if the transaction data is incorrect but it could also trigger an error value if the Payment Client generates a Java exception.

The response codes generated by the Payment Server are:

Code	Description
?	Response Unknown
0	Transaction Successful
1	Transaction Declined - Bank Error
2	Bank Declined Transaction
3	Transaction Declined - No Reply from Bank
4	Transaction Declined - Expired Card
5	Transaction Declined - Insufficient funds
6	Transaction Declined - Error Communicating with Bank
7	Payment Server Processing Error - Typically caused by invalid input data such as an invalid credit card number. Processing errors can also occur
8	Transaction Declined - Transaction Type Not Supported
9	Bank Declined Transaction (Do not contact Bank)
A	Transaction Aborted
C	Transaction Cancelled
D	Deferred Transaction
E	Issuer Returned a Referral Response
F	3D Secure Authentication Failed
I	Card Security Code Failed
L	Shopping Transaction Locked (This indicates that there is another transaction taking place using the same shopping transaction number)
N	Cardholder is not enrolled in 3D Secure (Authentication Only)
P	Transaction is Pending
R	Retry Limits Exceeded, Transaction Not Processed
S	Duplicate OrderInfo used. (This is only relevant for Payment Servers that enforce the uniqueness of this field)
U	Card Security Code Failed

---

## Card Type Code

The Card Type Code is a two-character field that identifies the card type that was used for the transaction.

Not all of these cards are available for all Payment Providers. Please check with your Payment Provider as to which cards you can use.

The Card Type Field values are:

Code	Description
AE	American Express
DC	Diners Club
JC	JCB Card
MC	MasterCard
VC	Visa Card

## Verified by Visa™ (Visa 3-Domain Secure) and MasterCard SecureCode™ (MasterCard 3-Domain Secure) Status Codes

All authentication transactions use a `vpc_VerStatus` response code value to show whether the card authentication was successful or not. The `vpc_VerStatus` response code values are:

Value	Description
Y	The cardholder was successfully authenticated.
E	The cardholder is not enrolled.
N	The cardholder was not verified.
U	The cardholder's Issuer was unable to authenticate due to a system error at the Issuer.
F	An error exists in the format of the request from the merchant. For example, the request did not contain all required fields, or the format of some fields was invalid.
A	Authentication of your Merchant ID and Password to the <b>Directory Server Failed</b> (see "What does a Payment Authentication Status of "A" mean?" on page 85).
D	Error communicating with the Directory Server, for example, the Payment Server could not connect to the directory server or there was a versioning mismatch.
C	The card type is not supported for authentication.
M	This indicates that attempts processing was used. Verification is marked with status M – ACS attempts processing used. Payment is performed with authentication.  Attempts is when a cardholder has successfully passed the directory server but decides not to continue with the authentication process and cancels.
S	The signature on the response received from the Issuer could not be validated. This should be considered a failure.
T	ACS timed out. The Issuer's ACS did not respond to the Authentication request within the time out period.
P	Error parsing input from Issuer.
I	Internal Payment Server system error. This could be caused by a temporary DB failure or an error in the security module or by some error in an internal system.

## Error Codes and their descriptions for the most commonly encountered errors

Error Number	Description
5001	Invalid Digital Order
5004	Invalid Digital Order: invalid session ID
5005	Invalid Digital Order: invalid Merchant Id
5006	Invalid Digital Order: invalid purchase amount
5007	Invalid Digital Order: invalid locale
5050	Invalid Permission
5061	Unsupported payment method
5065	Runtime exception
5121	Try to access an invalid key file
5134	RSA Decrypt Failed
5135	RSA Encrypt Failed
5231	Retrieved Digital Receipt Error
5423	Bad User Name or Password
5425	Invalid Recurring Transaction Number
5426	Invalid Permission
5433	Invalid Permission
5435	Max No of Deferred Payment reached
5436	Invalid recurring transaction number

## The complete list of Error Codes and their descriptions are:

Error Number	Description
5000	Undefined error
5001	Invalid Digital Order
5002	Invalid Digital Order: not enough fields
5003	Invalid Digital Order: too many fields
5004	Invalid Digital Order: invalid session ID
5005	Invalid Digital Order: invalid Merchant Id
5006	Invalid Digital Order: invalid purchase amount
5007	Invalid Digital Order: invalid locale
5008	Invalid Digital Order: outdated version
5009	<p>Invalid Digital Order: bad or too many Transaction Request parameters. It could be one of the following:</p> <ul style="list-style-type: none"> <li>▪ Invalid Digital Order: Invalid EBT Type Code field</li> <li>▪ Invalid Digital Order: Invalid PAN Entry Mode</li> <li>▪ Invalid Digital Order: Invalid PIN Entry Capability</li> <li>▪ Bad Credit Payment Type</li> <li>▪ Bad Account Balance Type</li> <li>▪ Unsupported Transaction Type</li> <li>▪ Invalid Digital Order: Invalid Payment Method</li> <li>▪ Invalid Digital Order: Invalid PIN field</li> <li>▪ Invalid Digital Order: Invalid KSN field</li> <li>▪ Invalid Digital Order: Invalid STAN field</li> <li>▪ Invalid Digital Order: Invalid PhysicalTerminalId field</li> <li>▪ Invalid Digital Order: Invalid POSEntryMode field</li> <li>▪ PIN Entry Capability Terminal Cannot Accept PIN</li> <li>▪ PIN Entry Capability Terminal PIN pad down</li> <li>▪ Authorisation Code must be provided</li> <li>▪ Authorisation Code must be numeric and 1 to 6 characters in length</li> </ul>

Error Number	Description
5020	Invalid Digital Receipt
5021	Invalid Digital Receipt: not enough fields
5022	Invalid Digital Receipt: too many fields
5023	Invalid Digital Receipt: invalid session ID
5024	Invalid Digital Receipt: invalid Merchant Id
5025	Invalid Digital Receipt: invalid purchase amount
5026	Invalid Digital Receipt: invalid locale
5027	Error in generating Digital Receipt ID
5028	Invalid Digital Receipt Delivery URL
5029	Invalid Digital Receipt Delivery IO
5030	Invalid Transaction log string
5031	Invalid Transaction log string: not enough fields
5032	Invalid Transaction log string: too many fields
5033	Invalid Transaction log string: invalid purchase amount
5034	Invalid Transaction log string: invalid locale
5035	Transaction Log File error
5040	Invalid QsiFinTrans message
5041	Unsupported acquirer
5042	Unsupported transport
5043	Unsupported message format
5044	Invalid Merchant transaction mode
5045	Unsupported transaction counter
5046	SecureCGIParam verification of digital signature failed
5047	Failed to read a QsiSigner object back from a serialized file!
5048	Failed to create a DCOM object
5049	Receipt is invalid.
5050	Invalid Permission
5051	Unsatisfied DLL link error
5052	Invalid Merchant Id
5053	Transmission error from QSIFinTrans
5054	Parser error
5055	Acquirer Response Error
5056	Trace file I/O error
5057	Invalid cookie
5058	RMI exception

Error Number	Description
5059	Invalid session
5060	Invalid locale
5061	Unsupported payment method
5065	Runtime exception
5066	Bad parameter name or value
5070	File backup error
5071	File save error
5072	File IO error
5073	File not found error
5074	File not found
5080	SQL Error
5081	SQL Error : Cannot locate the database
5082	SQL Error : Cannot connect to the database
5083	SQL Error : Incorrect row count
5084	SQL Error : Invalid value format
5085	SQL Error : Bad line count
5086	Duplicate primary agent
5087	Unknown database type
5090	Illegal user name
5091	Illegal password error
5101	Could not create and load the specified KeyStore object. If you are using a QSIDB KeyStore the database connection may have failed
5103	Could not create the specified javax.crypto.Cipher object. You may not have a provider installed to create this type of Cipher object or the Cipher object that is specified in your config file is incorrect
5104	Error in call to javax.crypto.Cipher.doFinal. Either the input was too large or the padding was bad
5106	The Message type specified is not supported. Check the com.qsipayments.technology.security.MessageCrypto.properties file to ensure that the MsgType is valid
5108	The message received has a bad format
5109	Error verifying signature
5110	Error creating a signature
5161	Customer Reference too long
5175	Card track data exceeded the allowed lengths
5120	Unable to generate new keys
5121	Try to access an invalid key file
5122	Not able to store the security keys
5122	Not able to store the security keys

Error Number	Description
5123	Not able to retrieve the security keys
5124	Encryption format invalid for Digital Order
5125	Encryption signature invalid for Digital Order
5126	Invalid transaction mode
5127	Unable to find user keys
5128	Bad key Id
5129	Credit Card No Decryption failed
5130	Credit Card Encryption failed
5131	Problem with Crypto Algorithm
5132	Key used is invalid
5133	Signature Key used is invalid
5134	RSA Decrypt Failed
5135	RSA Encrypt Failed
5136	The keys stored in the keyfile given to SecureCGIParam was corrupt or one of the keys is invalid
5137	The private key stored in the keyfile given to SecureCGIParam was corrupt or one of the keys is invalid
5138	The public key stored in the keyfile given to SecureCGIParam was corrupt or one of the keys is invalid
5140	Invalid Acquirer
5141	Generic error for a financial transaction
5142	Generic reconciliation error for a transaction
5143	Transaction counter exceeds predefined value
5144	Generic terminal pooling error
5145	Generic terminal error
5146	Terminal near full
5147	Terminal Full
5148	Attempted to call a method that required a reconciliation to be in progress but this was not the case
5150	Invalid credit card: incorrect issue number length
5151	Invalid Credit Card Specifications
5152	Invalid Credit Card information contained in the database
5153	Invalid Card Number Length
5154	Invalid Card Number
5155	Invalid Card Number Prefix
5156	Invalid Card Number Check Digit
5157	Invalid Card Expiry Date
5158	Invalid Card Expiry Date Length

Error Number	Description
5162	Invalid Card Initialisation file
5166	Invalid Credit Card: incorrect secure code number length
5170	Unable to delete terminal
5171	Unable to create terminal
5161	Customer Reference too long
5175	Card track data exceeded the allowed lengths
5176	Bad Card Track, invalid card track sentinels
5185	Invalid Acknowledgement
5200	Payment Client Creation Failed
5201	Creating Digital Order Failed
5202	Creating Digital Receipt Failed
5203	Executing Extension Command Failed
5204	Executing Administration Capture Failed
5205	Executing Administration Refund Failed
5206	Executing Administration Void Capture Failed
5207	Executing Administration Void Refund Failed
5208	Executing Administration Financial Transaction History Failed
5209	Executing Administration Shopping Transaction History Failed
5210	PaymentClient Access to QueryDR Denied
5220	Executing Administration Reconciliation Failed
5221	Executing Administration Reconciliation Item Detail Failed
5222	Executing Administration Reconciliation History Failed
5230	Retrieving Digital Receipt Failed
5231	Retrieved Digital Receipt Error
5232	Digital Order Command Error
5233	Digital Order Internal Error
5234	MOTO Internal Error
5235	Digital Receipt Internal Error
5336	Administration Internal Error
5337	Extension Internal Error
5400	Digital Order is null
5401	Null Parameter
5402	Command Missing
5403	Digital Order is null
5410	Unknown Field

Error Number	Description
5411	Unknown Administration Method
5412	Invalid Field
5413	Missing Field
5414	Capture Error
5415	Refund Error
5416	VoidCapture Error
5417	VoidRefund Error
5418	Financial Transaction History Error
5419	Shopping Transaction History Error
5420	Reconciliation Error
5421	Reconciliation Detail Error
5422	Reconciliation History Error
5423	Bad User Name or Password
5424	Administration Internal Error
5425	Invalid Recurring Transaction Number
5426	Invalid Permission
5427	Purchase Error
5428	VoidPurchase Error
5429	QueryDR Error
5430	Missing Field
5431	Invalid Field Digital.TRANS_NO must be provided to indicate which existing order this transaction is to be performed against
5432	Internal Error
5433	Invalid Permission
5434	Deferred Payment service currently unavailable
5435	Max No of Deferred Payment reached
5436	Invalid recurring transaction number
5440	Missing extension parameter
5441	ExtensionHandler: Invalid Recurring Transaction Number
5450	DirectPaymentSend: Null digital order
5451	DirectPaymentSend: Internal error
5500	Error in card detail
5501	Errors exists in card details
5600	Transaction retry count exceeded
5601	Instantiation of AcquirerController for this transaction failed.
5602	An I/O error occurred

Error Number	Description
5603	Couldn't get a valid terminal
5604	Unable to create the ProtocolReconciliationController for the protocol
5661	Illegal Acquirer Object Exception
5670	Message Exception
5671	Malformed Message Exception
5672	Illegal Message Object Exception
5680	Transport Exception
5681	Transport type not found
5682	Transport connection error
5683	Transport IO error
5684	Illegal Transport Object Exception
5690	Permanent Socket Transport connected
5691	Permanent Socket Transport JII class exception
5692	Permanent Socket Transport mismatched message received
5693	Permanent Socket Transport malformed message received
5694	Permanent Socket Transport unavailable
5695	Permanent Socket Transport disconnected
5696	The connection has been closed prematurely
5730	Host Socket unavailable
5750	Message header not identified
5751	Message length field was invalid
5752	Start of text marker (STX) not found where expected
5753	End of text marker (ETX) not found where expected
5754	Message checksum (LRC) did not match
5800	Init service started
5801	Init service stopped
5802	Invalid entry
5803	Duplicate entry
5804	Parse error
5805	Executing task
5806	Cannot execute task
5807	Terminating task
5808	Task killed
5809	Respawning task
5810	Cron service started

Error Number	Description
5811	Cron service stopped
5812	Parse error
5813	Invalid entry
5910	Null pointer caught
5911	URL Decode Exception occurred
5930	Invalid card type for excessive refunds
5931	Agent is not authorized to perform excessive refunds for this amount
5932	Too many excessive refunds apply to this shopping transaction already
5933	Merchant agent is not authorized to perform excessive refunds
5934	Merchant is not authorized to perform excessive refunds
5935	Merchant cannot perform excessive refunds due to its transaction type
6010	Bad format in Rulefile
6100	Invalid host name
7000	XML parser [Fatal Error]
7001	XML parser [Error]
7002	XML parser [Warning]
7003	XML Parameter is invalid
7004	XML Parameter had an invalid index. Check input .jhtml file
7005	XML [Bad Provider Class]
7050	SleepTimer: Time value is not in a valid format (ignored this time value)
7100	No valid times and/or interval specified in StatementProcessing.properties file. Execution terminated
7101	Status file for this data file was never created – deleting
7102	Error loading Statement.properties file
7104	Can't find file
7106	IOException thrown attempting to create or write to file
7107	Overwriting file
7108	SecurityException thrown when attempting to create output file
7109	Invalid Merchant Id. This Advice element will not be processed
7110	Can't create file name from the given date string
7111	Duplicate Advice element found in input document and skipped. Check input document
7112	Invalid payment type specified. This file will be skipped
7113	Null directory: can't create output file
7114	Validation of input file provided by host failed
7120	IOException thrown attempting to create or write to file
7121	IOException thrown while attempting to create a ZIP archive

Error Number	Description
7122	An inaccessible output directory was specified in the configuration file
7200	PRE Issue Id Error
7201	No Logon User Object stored in session.
7202	Error Occurred while creating the merchant on the Payment Server.
7203	Logging out
7204	Error occurred while instantiating Payment.
7205	Error occurred while instantiating SSL Payment
7207	Error occurred while sending email
7208	Invalid Access. User is trying to access a page illegally.
7209	Invalid User Input.
7300	Error parsing meta data file
7301	Invalid field
7302	Field Validator not present
7303	Validation of field failed
7304	Field not present in arbitrary data
7305	Mandatory field missing
7306	Date mask is invalid
7307	Error creating field validator
7308	Failed to update arbitrary data
7400	Invalid transaction type
7500	Record has changed since last read
8000	Invalid Local Tax Flag
8001	Local Tax Amount Equal to or Greater then Initial Transaction Amount
8002	Purchaser Postcode Too Long
8003	Invalid Local Tax Flag and Local Tax Flag Amount Combination
8004	Invalid Local Tax Amount
8015	Payment method must be EBT for a balance inquiry
8015	Invalid Digital Order: Invalid PaymentMethod
8016	Invalid Digital Order: Invalid PIN field
8017	Invalid Digital Order: Invalid KSN field
8019	Invalid Digital Order: Invalid PhysicalTerminalID field
8020	Invalid Digital Order: Invalid POEntryMode field
8021	Invalid Digital Order: Invalid AdditionalAmount field
9000	Acquirer did not respond

# Appendix – Test Environment

---

## Test Cards

The following table shows the test card numbers and associated expiry dates configured for each card scheme on the MIGS Payment Server.

Card Type	PAN	Expiry Date
MasterCard	5123456789012346	05/13
MasterCard	5313581000123430	05/13
Visa	4005550000000001	05/13
Visa	4557012345678902	05/13
Amex	345678901234564	05/13
Bankcard (Australian Domestic)	5610901234567899	05/13
Diners Club	30123456789019	05/13

---

## Issuer Response Code Mapping

The Payment Server returns both a summary result code generated by the Payment Server as well as the raw issuer response code as received from the bank.

The following table is a list of relevant issuer response codes:

Issuer Resp.	Description
00	Approved
01	Refer to Card Issuer
02	Refer to Card Issuer
03	Invalid Merchant
04	Pick Up Card
05	Do Not Honor
07	Pick Up Card
12	Invalid Transaction
14	Invalid Card Number (No such Number)
15	No Such Issuer
33	Expired Card
34	Suspected Fraud
36	Restricted Card
39	No Credit Account
41	Card Reported Lost
43	Stolen Card
51	Insufficient Funds
54	Expired Card
57	Transaction Not Permitted
59	Suspected Fraud
62	Restricted Card
65	Exceeds withdrawal frequency limit
91	Cannot Contact Issuer

The following table shows how the bank simulator maps the issuer response code to response codes.

Issuer Resp.	MIGS Resp.	Issuer Resp.	MIGS Resp.	Issuer Resp.	MIGS Resp.
00	0	34	2	68	3
01	2	35	1	69	1
02	2	36	2	70	1
03	2	37	1	71	1
04	2	38	1	72	1
05	2	39	2	73	1
06	2	40	1	74	1
07	2	41	2	75	1
08	0	42	1	76	1
09	1	43	2	77	1
10	1	44	1	78	1
11	1	45	1	79	1
12	1	46	1	80	1
13	1	47	1	81	1
14	2	48	1	82	1
15	2	49	1	83	1
16	0	50	1	84	1
17	1	51	5	85	1
18	1	52	1	86	1
19	2	53	1	87	1
20	1	54	4	88	1
21	1	55	1	89	1
22	1	56	1	90	2
23	1	57	1	91	2
24	1	58	1	92	2
25	2	59	2	93	1
26	1	60	1	94	1
27	1	61	2	95	1
28	1	62	1	96	1
29	1	63	1	97	1
30	1	64	1	98	2
31	2	65	2	99	2
32	1	66	1		
33	4	67	1		