



## **ePDQ Cardholder Payment Interface (CPI)**

### **Integration Guide**

**V6.0 Released August 2007**

**Software Version: 5.9 ePDQ Payment Engine & Internet Authentication**

#### **COPYRIGHT NOTICE**

No part of this publication may be reproduced, stored in a retrieval system or transmitted in any form or by any means electronic or mechanical, including photocopying and recording, for any purpose, without the prior written permission of Product Development, Barclaycard Business, Barclays Bank PLC.

## Doc Version Control

Version No.	Date Issued.	Reason for Change
2.0	July 05	New URL for configuring ePDQ
		Internet Authentication Transaction Response- payer security level of 4 amended to reflect a MasterCard ECI value of 1
		Card Scheme rules changes for Switch, Solo and Maestro cards
		Changes to the allowed URL
		Footnote added to PHP script
		ASP.net example script
3.0	June 06	ePDQ CPI configuration – Post email
4.0	October 06	MasterCard SecureCode liability shift revisions
		Revised Contact Times
5.0	March 07	Amendment to 'Fixed price sites' example
6.0	August 07	Maestro changes

# Contents

CONTENTS.....	3
PURPOSE OF THIS DOCUMENT.....	5
AUDIENCE .....	5
CONTACTING US.....	5
GLOSSARY .....	6
INTRODUCTION .....	7
INTERNET AUTHENTICATION .....	8
<i>Registration</i> .....	8
<i>Deactivating Internet Authentication</i> .....	8
<i>Unable to process Internet Authentication transactions</i> .....	9
EPDQ INTEGRATION USING A TYPICAL EXAMPLE.....	10
<i>Encryption</i> .....	11
<i>Send Data</i> .....	11
<i>Receive Data</i> .....	12
DIRECTING CUSTOMERS TO THE EPDQ CPI CHECKOUT.....	13
CPI SOFTWARE / HARDWARE REQUIREMENTS.....	13
INTERNET AUTHENTICATION REQUIREMENTS .....	13
CPI SECURITY REQUIREMENTS .....	13
GETTING STARTED.....	14
STEP A: SET YOUR OWN USERNAME AND PASSWORD .....	14
STEP B: SIGN ON TO THE EPDQ CONFIGURATION PAGES .....	15
STEP C: EPDQ CPI CONFIGURATION .....	15
<i>Passphrase</i> .....	16
<i>Allowed URL</i> .....	16
<i>Continue Option 1/2/3 flags</i> .....	17
<i>POST Order Result</i> .....	18
<i>POST URL</i> .....	18
ENCRYPTING THE PRICE TO PAY AND THE ORDER DATA .....	19
INTEGRATION.....	19
MANDATORY CPI INFORMATION.....	20
EXAMPLE WEB PAGE .....	22
FIXED PRICE SITES.....	22
PERIODIC ORDERS (RECURRING BILLING) .....	23
HANDLING THE RETURNED RESPONSE.....	24
ORDER TRACKING .....	24
PROGRAM REQUIREMENTS (RESPONSE HANDLING PROGRAM INVOKED BY THE POST URL).....	25
<i>Transaction Status Results</i> .....	26

<i>Cardholder Transaction Status Message</i> .....	27
<i>Internet Authentication Transaction Response</i> .....	28
<i>Suggested Test Procedures</i> .....	29
<i>Internet Authentication Testing</i> .....	30
TRANSACTION RECONCILIATION .....	30
<b>PUTTING YOUR EPDQ STORE “LIVE” .....</b>	<b>31</b>
<b>TROUBLE SHOOTING .....</b>	<b>32</b>
<i>Error Messages</i> .....	32
<i>Declined Transactions</i> .....	33
<i>PostURL Problems</i> .....	33
<b>FREQUENTLY ASKED QUESTIONS .....</b>	<b>34</b>
<b>CONTACTING THE SUPPORT TEAM .....</b>	<b>37</b>
<b>APPENDICES: SAMPLE SCRIPTS.....</b>	<b>38</b>
<b>APPENDIX A: ASP .....</b>	<b>39</b>
ASPTear ENCRYPTION SCRIPT (ASP_TEAR_EXAMPLE.ASP) .....	39
<b>APPENDIX A: ASP (CONTD.).....</b>	<b>40</b>
MICROSOFT XML PARSER (SERVERXMLHTTP) ENCRYPTION SCRIPT (ASP_XML_EXAMPLE.ASP).....	40
<b>APPENDIX A: ASP (CONTD.).....</b>	<b>41</b>
EPDQ RETURNED RESPONSE HANDLING SCRIPT (ASP_RESPONSE_EXAMPLE.ASP).....	41
<b>APPENDIX B: ASP.NET .....</b>	<b>42</b>
ASP.NET ENCRYPTION SCRIPT .....	42
<b>APPENDIX B: ASP.NET (CONTD.) .....</b>	<b>44</b>
EPDQ RETURNED RESPONSE HANDLING SCRIPT (ASP.NET_RESPONSE_EXAMPLE) .....	44
<b>APPENDIX C: PHP.....</b>	<b>45</b>
ENCRYPTION SCRIPT (PHP_ENCRYPTION_EXAMPLE.PHP) .....	45
<b>APPENDIX C: PHP (CONTD.) .....</b>	<b>47</b>
EPDQ RETURNED RESPONSE HANDLING SCRIPT (PHP_RESPONSE_EXAMPLE.PHP) .....	47
<b>APPENDIX D: PERL.....</b>	<b>48</b>
ENCRYPTION SCRIPT .....	48
<b>APPENDIX D: PERL (CONTD.) .....</b>	<b>50</b>
EPDQ RETURNED RESPONSE HANDLING SCRIPT.....	50
<b>APPENDIX E: COMMON RESPONSE CODES .....</b>	<b>52</b>

## Purpose of this Document

This document is designed to provide our Internet Merchants with the relevant instructions and HTML code to integrate into their storefront, which communicates with the ePDQ secure Cardholder Payment Interface. It also provides details on how the ePDQ CPI handles transactions that can be authenticated under the Internet Authentication service.

## Audience

This document is intended for use by:

- Merchants integrating the ePDQ CPI
- Merchant Development Partners
- Web Developers

A working knowledge of HTML and an understanding of scripting and programming languages are required to integrate ePDQ and understand this document.

You must ensure that you have experience with the skill sets required, or that you use a developer or Merchant Development Partner to provide these skills. All of the ePDQ Merchant Development Partners are listed on the main ePDQ web site at:-

[http://www.barclaycardbusiness.co.uk/information\\_zone/products/epdq\\_preferred\\_partners.html](http://www.barclaycardbusiness.co.uk/information_zone/products/epdq_preferred_partners.html)

## Contacting us

Contact us on 0870 60 80 355\*

Monday to Friday:	8.00am to 8.00pm
Saturday:	9.00am to 5.00pm.

Alternatively you can email us at: [ecomm.support@barclaycard.co.uk](mailto:ecomm.support@barclaycard.co.uk)

***\* Calls may be monitored and recorded.***

## Glossary

Term	Definition
ASP	Active Server Page – Microsoft’s Server side scripting language.
CGI	Common Gateway Interface – a way of interfacing computer programs
CPI	Cardholder Payment Interface – secure checkout payment page
ePDQ	Internet payment solution from Barclaycard Business
GBP	Great British Pound (Sterling)
ISO	A recognised protocol for transaction transmission
HTML	Hypertext Mark up Language – language used to construct web pages
HTTP	HyperText Transfer Protocol – the protocol used most often to transfer information from World Wide Web servers to browsers. Also called HyperText Transport Protocol
HTTPS	HyperText Transfer Protocol, Secure. A version of http for secure transactions
OID	Order ID. Unique reference associated to a transaction
PERL	Practical Extraction and Reporting Language - A general-purpose programming language
PHP	Hypertext Preprocessor – Open Source Server side scripting language
POST	An HTTP command which sends data from a client to a server, or one server to another
SecureCode	A MasterCard and Maestro initiative to authenticate cardholders for online purchases
SSL	Secure Sockets Layer – a protocol designed to provide secure communications on the internet
URL	Uniform Resource Locator – an internet address
VbV	Verified by Visa. A Visa initiative to authenticate cardholders for online purchases

## Introduction

This document describes minimum requirements to allow ePDQ to process both authentication (for Internet Authentication – see below) and authorisation transactions in real time via the ePDQ Cardholder Payment Interface (CPI).

The ePDQ CPI is an HTML based integration. It uses standard form information to submit a number of variables to a Barclaycard Business securely hosted payment page for the purposes of transaction processing. Once processed, transaction status information is returned to a predefined script to allow fulfilment of orders automatically.

To enhance the security of transaction information and responses, ePDQ employs encryption modules (ASP or Perl) to prevent tampering which you install on your web servers.

ePDQ provides a secure CPI checkout to:

- present a payment page to your customers requesting card and billing / delivery details
- submit card information to facilitate cardholder authentication using Verified by Visa and SecureCode for MasterCard and Maestro cards
- submit transactions for real-time authorisation
- present order confirmations and return status results
- return customers to your store

If you wish, you can pre-populate some of the payment page details and suggest a preferred colour scheme/logo to help maintain branding. For further details please see the accompanying document titled ePDQ Cardholder Payment Interface Integration Enhancements (for v5.9 ePDQ Payment Engine and Internet Authentication).

Additional reference should be made to the User Guide for further information on the features of ePDQ which includes a section on 'Risk Management'.

## Internet Authentication

Internet Authentication allows you to check the authenticity of a cardholder when they choose to purchase from your website. The service is only applicable to Visa transactions using Verified by Visa, and MasterCard and Maestro transactions using SecureCode

The ePDQ CPI automatically checks to see if authentication is possible on every transaction. If it is, the CPI will show your customer an 'in-line window' supplied by their card issuer which asks them to authenticate themselves. If they are able to authenticate themselves, a value is attached to the transaction, which is then sent for authorisation.

Even if the cardholder is not registered to authenticate themselves, we will attach a value to the transaction to ensure that the card issuer knows you tried to authenticate the cardholder.

There may be occasions when the cardholder is unable to authenticate themselves. If they fail to authenticate then the transaction will be stopped as you cannot guarantee that the cardholder is who they say they are. If they are unable to authenticate due to a system error, then you have the choice of whether to proceed with the transaction or not.

You must read the Internet Authentication Procedure Guide to fully understand how Internet Authentication works.

### Registration

Before you can use the Internet Authentication service, we must register you with the participating Card Schemes (Visa, MasterCard and Maestro). You will be allocated a unique identifier for Visa. Visa use the identifiers to validate your identity each time you submit a transaction. We will also integrate this identifier into your ePDQ configuration so that you are recognised each time you send transaction data to ePDQ.

The registration process can take up to 10 working days. We will start the registration process as soon as your application is approved. You will not be able to go fully live with Internet Authentication until we have advised you that you are fully set up.

You could however trade without Internet Authentication before this time. If you do this you will **not benefit** from charge back liability shift.

### Deactivating Internet Authentication

If you choose not to use the Internet Authentication service supplied by the ePDQ CPI, you must notify us in writing (an email will be sufficient). We will deactivate Internet Authentication from your ePDQ CPI.

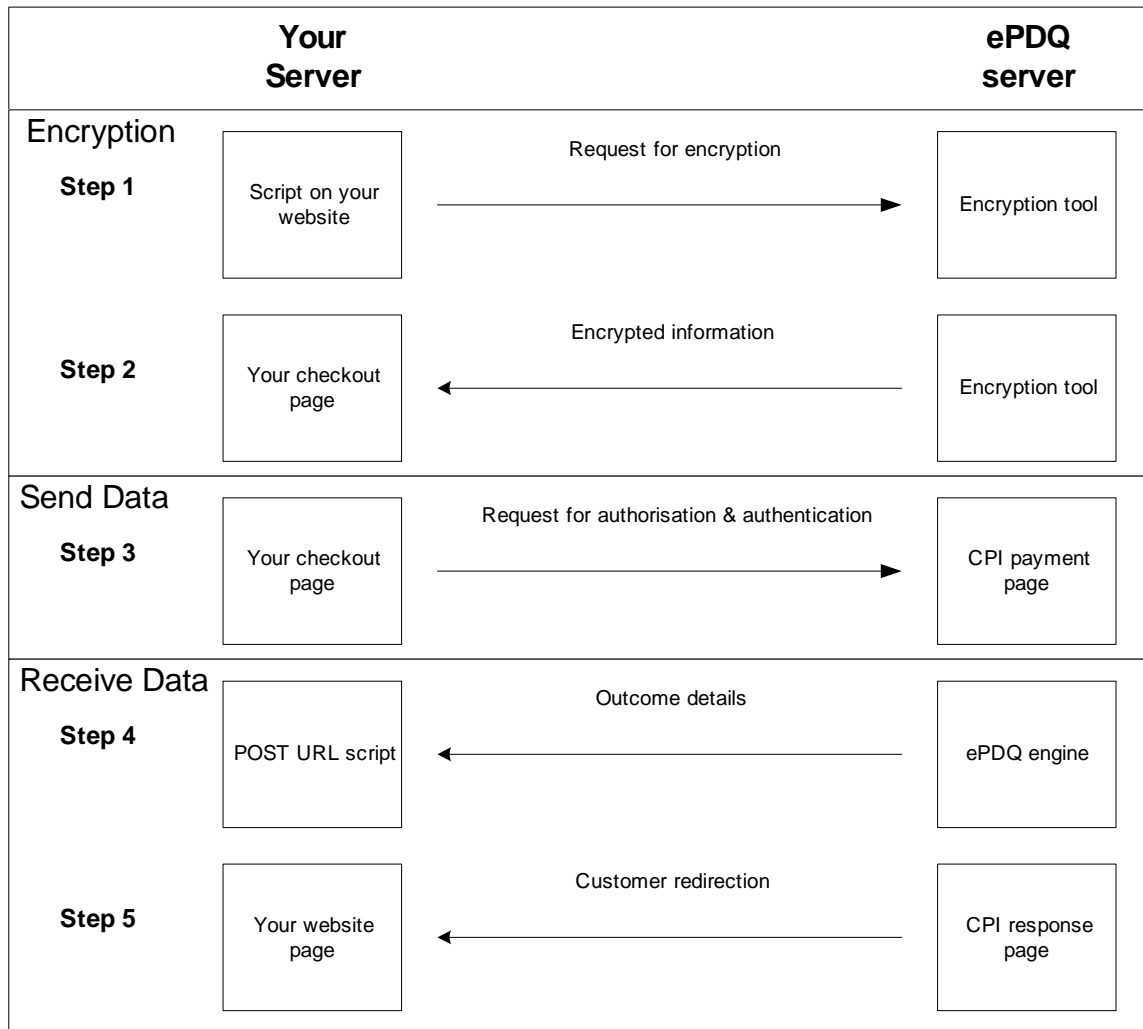
You must be aware that you will no longer be covered by the chargeback protection offered by Internet Authentication should you request deactivation. The CPI will not prompt the cardholder to authenticate themselves. In the event of a challenged transaction you will be charged back.

Requests for deactivation should be sent to the eCommerce Support Team. See 'Contacting Us' for further information.

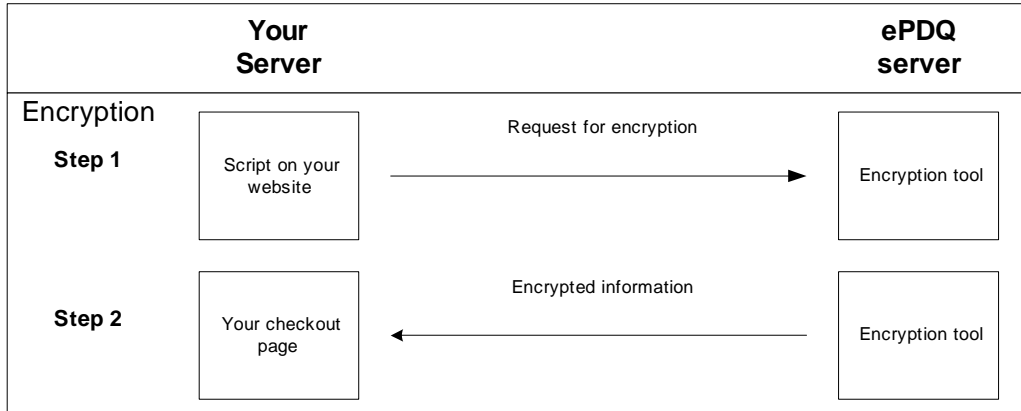
## **Unable to process Internet Authentication transactions**

The Internet Authentication service has a number of technical dependencies as it involves merchants, issuers and the card Schemes Visa, MasterCard and Maestro. There may be occasions where the service is not available. On these occasions, an error will be returned, and the CPI will treat the transaction as a normal transaction without authentication (see Section on Continue Option 1/2/3 flags for how you can handle this). If you continue processing the transaction after an error has occurred you will lose liability shift.

## ePDQ integration using a typical example



## Encryption



### Step 1

Your customer browses your website, fills their shopping basket and clicks to go to the checkout. The CPI process starts here. A script on your website makes the first call to the encryption server, requesting the encryption of the transaction details. The transaction details include the total price to pay and the order details.

**Note:** If invalid data is provided the transaction will not process correctly.

### Step 2

The encryption tool encrypts the transaction data. This generates an HTML <INPUT> tag containing the encrypted transaction data referred to as epdqdata string.

**Note:** This HTML needs to be “pulled” back from the epdq server; it will not be posted to you.

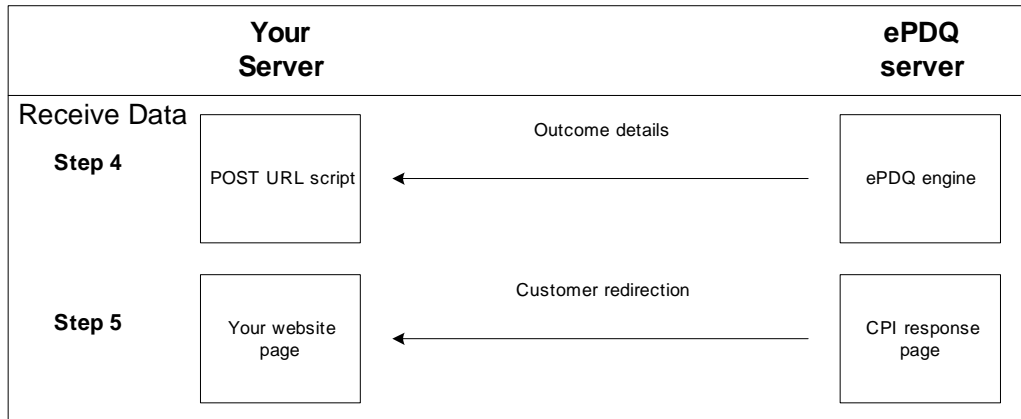
## Send Data



### Step 3

Your script then generates a HTML form. This must include the returned epdqdata string as a variable and the remaining mandatory variables. In addition you may include in the form any extended variables you wish to use. This form is submitted to the CPI payment page. Any extended variables that are included in the form pre-populate the payment page. This avoids the cardholder inputting the data twice, once on your website and once on the payment page.

## Receive Data



### Step 4

Your customer completes the remaining fields on the CPI payment page, including their credit card number. Once the customer chooses to process with the transaction it is submitted for processing. The customer will be asked to Authenticate at this stage if appropriate. Once the transaction is processed, the transaction status response and other details of the transaction are posted back to your pre-determined URL. This data allows you to fulfil the transaction and update the outcome of the transaction on your system.

### Step 5

The CPI generates a response page which tells the cardholder what the outcome of the transaction was –approved or declined. If the response is a decline there may be a reason displayed to the cardholder. This page has a Continue button which once clicked on redirects the customer back to your website.

The Order ID is also sent across to your website to allow you to identify the customer and link the customer to the detailed information sent in Step 4.

## **Directing Customers to the ePDQ CPI Checkout**

ePDQ has been designed to integrate into your online store by use of our Cardholder Payment Interface (CPI) to provide a secure checkout environment.

This permits standard http (non-SSL) HTML web pages on your site to call our secure service for payment authorisation.

Please be aware that if you present the secure CPI page within frames on your web site, the browser may not display the recognised 'https' or 'padlock' indicating a secure page. Some cardholders may decline to enter card details if they do not see these widely known symbols.

## **CPI Software / Hardware Requirements**

The ePDQ CPI does not impose any direct restrictions on your choice of storefront software or hardware beyond that required to POST to ePDQ and to (optionally) accept returned CGI data.

## **Internet Authentication Requirements**

The Internet Authentication software is fully integrated into the ePDQ CPI. To set-up the CPI you must make three choices around when to proceed or reject transactions when authentication is not possible (due to system or cardholder error). You must ensure that we have confirmed your registration with Visa and MasterCard before you attempt to process an Internet Authentication transaction.

## **CPI Security Requirements**

ePDQ has been designed to present a secure online checkout for your customers, and remove the need for you to design and host your own payment page.

In order to help maintain a secure connection between ePDQ and your storefront a number of security measures are in place. These have been especially designed for ePDQ to provide a high level of protection.

To ease integration we will describe these in two phases: how to supply the price to ePDQ and how to handle the response from ePDQ. Obviously, you should tailor these examples to meet your own specific requirements.

Before integration begins you will need to complete the CPI Administration Tool, and configure your ePDQ account. Instructions on how to configure your store, set-up Internet Authentication processing and log onto the CPI Administration Tool are provided on the following pages.

## Getting Started

We need to know some basic information about how you want to operate the link between your storefront and the ePDQ CPI (Cardholder Payment Interface). This section gives you a step by step guide to providing the information we require.

### Step A: Set your own Username and Password

Barclaycard Business provides you with a user account to access the Store Admin. Your Store Admin user role is set at ePDQ Level 4. We recommend that you use a separate user account to integrate the ePDQ CPI, with a user role of CPI Access. This allows you to control the permissions of the person integrating the CPI and prevents access to the Store Admin.

**NOTE for Developers: If you are integrating ePDQ on behalf of a Barclaycard Business customer, please contact the customer to obtain the CPI Access user details. This will allow you to access the CPI Administration Tool covered in the following steps.**

1. If you do not already have your Store Admin password login details please contact our eCommerce Support team on 0870 60 80 355.
2. Log into your ePDQ store, click **Administration** from the top four options
3. Select **Users** from the menu on the left
4. From the **User Configuration Page** (Please see the ePDQ User Guide section “Controlling Access to your Store for details), select **Add**.
5. The **Add User** page will be displayed. Enter the details as required.
  - The **User ID** must be at least 8 alpha/numeric characters and can be up to 32 characters. It must not contain any special characters (such as \*,/, \, -,). A User ID such as CPIUser1 is recommended.
  - The **Password** must be at least 8 alpha/numeric characters and must not contain any special characters.
  - Enter the **Password** again for validation purposes.
  - Enter a **Password Prompt**. This must not be your password and must not compromise the password (i.e. “same as user ID”).
  - **Account Name** and **Account Description** are optional, though entering a name and description such as “CPI Integration User” and “User for CPI Access only” is recommended.
  - The **Role** is the level of permissions you wish to assign to the user. Use the drop down list to select the role of CPI Access.
  - Set **Pagination** size according to how much data you wish to have displayed on each page.
6. You should elect for an email to be sent to the merchant confirming the creation of a new user.
7. Once you have entered all the details correctly, select **Add**. If you have entered any details incorrectly and wish to clear all fields, select **Reset**.
8. The new user will now be created, and can be viewed from the **User Configuration Page**.
9. Log on as the new user to confirm the user had been created correctly.
10. Sign off from the engine.

**IMPORTANT - If you have enabled the password security policy within your ePDQ store you must ensure that the specific CPI user and password never expires. Always select the ‘Password never expires’ option to avoid this user and password expiring.**

This user will be required for the “behind the scenes” transaction processing performed by the CPI and must be maintained at all times (it must not be deleted or modified).

You can now proceed to step B.

## Step B: Sign on to the ePDQ Configuration Pages

On your normal web browser please visit the following web site URL:

<https://cpiadmin.epdq.co.uk/cgi-bin/CcxBarclaysEpdqAdminTool.e>

Enter your assigned ePDQ Client Id (Store ID), and the Username and Password that you have just created for the 'CPI Access' user account by following the 'Logging In' instructions in Step 1. You can now go to STEP C.

## Step C: ePDQ CPI Configuration

When you have signed in correctly you will be presented with a form where you can enter the following details.

Field	Description
Client Id User Name Password	<p>These fields must contain the same details that you used to sign into the ePDQ Configuration service (Step B). Once you have successfully signed on the Client ID and User name should already be populated but the Password field will remain blank. Please check to ensure that the Client ID displayed is the correct one you wish to configure.</p> <p>If these details are not displayed then you must configure them as follows:</p> <p><b>Client ID:</b> This must be the client ID of the ePDQ store you wish to configure and match the Client ID used at the sign-on (Step B)</p> <p><b>Username and Password:</b> Must match a valid account set up in the 'Administration' section of the ePDQ Store Admin. It must also be the same account used at the sign-on (Step B)</p> <p>The User account must be maintained in order to allow the CPI to function. If this account is deleted or modified, the CPI will no longer function. (Client Id is also known as your Store ID)</p>
Passphrase	<p>The passphrase is a value which you create. You need to include this value in the encryption process as described on page 20 '<i>Encrypting the price to pay and the order data</i>'. The field name for this value in the encryption script is 'Password'.</p> <p>Typically this would be hard coded into your encryption script as the value for the fieldname 'password'. This can be a phrase or string, up to 16 characters.</p> <p>ePDQ will lookup and match your passphrase to the password above recorded for your store for validation purposes. (See note below).</p>
Allowed URL	<p>The full address (including http://) of the page on your web site, which submits the mandatory variables to the CPI payment page.</p> <p style="text-align: right;">Size: up to 127 characters</p>
Continue Option 1	<p>An explanation of this is on the next page. The default is YES.</p>

Field	Description
Continue Option 2	An explanation of this is on the next page. The default is YES.
Continue Option 3	An explanation of this is on the next page. The default is YES.
POST Order Result	For ePDQ to send your storefront the results of transactions in real-time please set this to "Yes".
POST URL	The full address (including http://) of the transaction fulfilment script on your web site. This MUST be in lower case to avoid browser case sensitivity problems, and must not involve port numbers or be an IP address. This must be a standard non-SSL (http://) not an SSL address (https://). Size = 128 characters [Only required if POST order Result is set to "Yes"]
POST User name POST Password	Username and Password that you have assigned to protect your CGI program (see Program Requirements for info). [Only required if POST order Result is set to "Yes"]
POST Email	Enter an email address here if you need to be informed of any failures with the Post URL process

### Passphrase

The passphrase parameter allows you to pass a value other than your store username or password to the CPI encryption, removing the need for you to share your store password. When submitted, the ePDQ engine uses the passphrase to match to your store username and password to validate your identity when processing transactions.

### Allowed URL

ePDQ needs to know the full web site address (URL) from where you will be calling the ePDQ payment page, this is the Allowed URL.

Please note that the "Allowed URL" must be the full, fixed, URL that is actually submitting the variables, determined by right mouse clicking on the page and viewing the page properties. This validates that the request came from your web site.

It can be from a standard non-SSL (http://) page or SSL (https://) page. The CPI validates that the request originates from your website by checking the HTTP\_REFERER value in the environment variables sent across with the customer. Please note, certain older browsers may actually block the HTTP\_REFERER value being sent – if your customers are using one of these older browsers, and your website is SSL protected, the customer may experience a 'Not a valid Allowed URL' error message. This is for privacy concerns as documented in the Microsoft © knowledge-base article number Q178066.

Finally, if your page appends varying data (for example "?customer=1234") to the URL, then you may need to use a jump page on your web site to redirect requests to ePDQ. These can use JavaScript to automatically submit a redirected form as follows:

```
<body onLoad="document.forms[0].submit()">
```

**Note:** A **jump page** is a self submitting form, typically generated by a script, with the source HTML being created on the fly based on the individual transaction details.

## Continue Option 1/2/3 flags

Typically, if a cardholder is registered to use the Internet Authentication service, they will be able to authenticate themselves at the time of the transaction. There may, however, be occasions where authentication is not successful.

Authentication may fail for various reasons, for example:

1. If the cardholder fails to correctly key in their password after three attempts, or
2. If the in-line window times out, or
3. Failure to provide the in-line window through connectivity failure.

Each of these scenarios has an impact of the liability shift. There are three flags which control what happens in the failure cases.

If the cardholder fails to correctly key in their password for Verified by Visa transaction the transaction is automatically declined. This is part of the Visa scheme regulations and you can not choose to continue with the transaction with the same card number.

### Continue Option 1

This flag controls the outcome of processing when any form of VbV error occurs or there is an error with the CPI.

The cardholder may avoid authenticating themselves by allowing it to timeout. This may be because they are not familiar with the in-line window, they may have forgotten their password, or they are deliberately avoiding authenticating themselves. Typically the card issuer will treat this as a failed authentication. If this occurs, the CPI will automatically decline the transaction. If the issuer does not treat this as a failed authentication and allows processing to continue you may choose to proceed with the transaction and must be aware that you will **lose the protection** afforded by the chargeback liability shift

Other errors may prevent VbV processing occurring such as failure to provide the in-line window through connectivity failure. This also means you **lose the protection** of the liability shift.

If a failure occurs within the CPI you **lose the protection** of the liability shift. This applies to both MasterCard and Visa transactions.

If you wish to accept the risk of processing a transaction for the examples above you should set the 'Continue Option 1' to YES (default).

If you do not want to accept the risk and you want to decline ALL transactions where VbV authentication or the CPI fails you should set the 'Continue Option 1' to NO.

You will **lose the protection** of the liability shift on these transactions if you do choose to process these transactions.

## Continue Option 2

This flag controls the outcome of processing when there is a failure with the SecureCode service. This covers technical failures and if the cardholder avoids authenticating themselves by allowing it to timeout. This may be because they are not familiar with the in-line window, they may have forgotten their password, or they are deliberately avoiding authenticating themselves.

If you wish to accept the risk of processing a MasterCard or Maestro transaction for the example above you should set the 'Continue Option 2' to YES (default).

If you do not want to accept the risk and you want to decline MasterCard or Maestro SecureCode transactions where these failures occur you should set the 'Continue Option 2' to NO.

You do **not lose the protection** afforded by the liability shift by processing these transactions.

## Continue Option 3

This flag controls the outcome of processing for MasterCard and Maestro SecureCode transactions when the cardholder fails authentication. This means the issuer has determined that the cardholder has incorrectly entered their information. This may mean the cardholder has forgotten their password or it may not be the genuine cardholder attempting this transaction.

If you wish to accept the risk of processing a transaction for MasterCard and Maestro SecureCode where the cardholder fails authentication you should set the 'Continue Option 3' to YES (default).

If you do not want to accept the risk and you want to decline ALL MasterCard and Maestro SecureCode transactions where authentication fails you should set the 'Continue Option 3' to NO.

You do **not lose the protection** afforded by the liability shift by processing these transactions.

## POST Order Result

If you choose not to use the POST option, then you will have to rely on checking either your Store Admin tool or use the transaction emails returned to you for each order.

**Note:** You will not receive 'real time' transaction results.

## POST URL

Transaction result data is posted to a predefined fulfilment script hosted on your web server – this is the POST URL. This must be a standard non-SSL page (<http://>).

Note **the POST URL and RETURN URL would typically be different**. According to recommended practices they should also be in lower case.

The **RETURN URL** is the page on your website the cardholder will be redirected to.

The POST to the CPI and the display of the confirmation page are completely independent. Only a subset of the data is posted back to your storefront (RETURN URL) directly.

If you do not have the required information to hand then you can always amend the details at a later time.

## Encrypting the price to pay and the order data

### Step 1 and 2

The price to pay (total amount to be charged to the cardholder including shipping and VAT) and order data must be supplied encrypted by your storefront to ePDQ in order to help prevent any potential price tampering. The encryption is carried out by ePDQ [Step 1]. Your storefront will then need to pull the encrypted data back [Step 2].

To ease integration your storefront should, as it prepares the final “pay now” form, ask ePDQ to encrypt the details for you. You can simply incorporate the returned, encrypted details within the “pay now” form.

This documentation contains several example scripts that can perform the encryption process discreetly, typically being used on your site when the customer is ready to finalise their transaction. Please refer to the appendices at the back of this manual, and download the relevant.ZIP file from <http://www.epdq.co.uk/nextsteps/cpi.htm>

ePDQ expects the following information for encryption:

Variable	VALUE
Total*	total order value
oid	order number
Chargetype*	immediate or delayed shipment
Password*	ePDQ administration service passphrase
Currencycode*	ISO currency code (e.g. 826 for gbp)
Clientid*	ePDQ administration service Client Id (Also known as your Store ID)

Where \* indicates a mandatory field.

**Please note, these values must not be passed as HTML form values, they must be encrypted.**

There is no additional data required to generate an Internet Authentication transaction. However, if you fail to provide a valid store ID and password we will be unable to validate your details and perform the authentication for you.

## Integration

Precise instructions depend upon your choice of web server and storefront application.

We have included instructions for use with ASP, PHP and Perl as these are amongst the most widely used computer languages on the Internet. These are intended as examples only and should be used as a basis to understand how to integrate ePDQ. Similar techniques can also be adopted with other computer languages. Please see the appendices attached to this document for more information.

## Mandatory CPI Information

### Step 3

Your web site must then supply the following mandatory information to ePDQ [Step 3]. Failure to supply this information will result in an error message being generated.

Key	Value
epdqdata	Sensitive information (such as price and transaction type) that your storefront has asked ePDQ to encrypt for added security (see below)
returnurl	Full URL of standard HTML web page that your customers are directed to upon completion of the transaction Size: Up to 100 characters
merchantdisplayname	Your trading name that will be displayed on the ePDQ payment page and order status page. This is a Card Scheme requirement. If you are using Internet Authentication the cardholder in-line box only displays the first 25 characters of your merchant name. If this causes an issue you may wish to shorten your merchant name to less than 25 characters. Size: Up to 30 characters

Where **epdqdata** contains the following encrypted information:

Key	Value
clientid	Barclaycard Business assigned numeric identifier for your store. (Also known as your Store ID) Size: Up to 20 digits
oid	Text string that can contain the unique order reference code to use for this transaction. Oid must be alpha/numeric only (no special characters other than dash).  The CPI will not accept an unencrypted order ID. If you choose not to pass the Order ID for encryption, ePDQ will generate one for you. It is strongly recommended you supply an order ID for tracking and reconciliation purposes Size: Up to 36 alphanumeric characters
password	Text string containing your <u>passphrase</u> . Important – this must be the passphrase and not the store password. Size: Up to 16 characters
chargetype	An indicator of whether the store provides immediate or delayed shipment. Possible values are:  Auth (immediate shipment) <small>see Note 1</small> PreAuth (delayed shipment) <small>see Note 2</small>  Note: these are case sensitive and must appear as shown
currencycode	ISO numeric currency code that must match that assigned to your store. Typically “826” for sterling transactions (see ‘CPI Integration Extensions’ document for information on additional currencies) Size: 3 digits

Key	Value
total	Total numeric value of purchase, including all items, tax and delivery charges, as generated by your storefront  Range: 1.00 to 999999999.00. Size: Up to 12 characters

Note 1 ePDQ is able to authorise and settle every transaction automatically. You must ensure that if using this option you are able to despatch goods immediately, or obtain cardholder agreement for any delay.

Note 2 If your business is unable to despatch goods immediately, you may wish to use this option. If you do operate the pre-auth model, then you must ensure that you mark the transactions for settlement when you despatch the goods. Failure to do this may result in funds not being paid to your account. Please be aware that some card issuers will remove the authorisation from a transaction after a certain period of time has elapsed. You must ensure that you despatch and mark transactions for settlement as soon as possible via the Store Admin Tool.

## Example Web Page

This example demonstrates the minimal information required by the ePDQ CPI. The encrypted ePDQ code should be inserted using one of the techniques shown on the following pages.

```
<html>
<head><title>My Store</title></head>
<body>
<form action="https://secure2.epdq.co.uk/cgi-bin/CcxBarclaysEpdq.e" method="post">

<!-- place encrypted ePDQ code between the form tags -->

<input type="hidden" name="returnurl" value="http://www.store.com/thankyou.html">
<input type="hidden" name="merchantdisplayname" value="My Store">
<input type="submit" value="purchase">
</form>
</body>
</html>
```

## Fixed price sites

Sites that have a fixed price per sale can “hard-code” the pre-encrypted details into their web page. Typically this can be used in place of a shopping basket.

Please note the order number must not be hard-coded since it must be unique for each order. In this case simply do not supply an order number when generating the encrypted data, allowing ePDQ to generate a unique code for you. Please remember that if you do not supply an order ID from your storefront, it may be more difficult to track the order through your system.

To generate the fixed, encrypted, data you can incorporate the following form into a temporary web page on your local computer...

```
<form action="https://secure2.epdq.co.uk/cgi-bin/CcxBarclaysEpdqEncTool.e" method="post">
<p>Auth Type: <input type="text" name="chargetype" value="Auth or PreAuth">
<p>Client Id: <input type="text" name="clientid" value="Your Client Id">
<p>Password: <input type="text" name="password" value="Your passphrase">
<p>Currency: <input type="text" name="currencycode" value="826 (For GBP Stores)">
<p>Total: <input type="text" name="total" value="1.00">
<p><input type="submit" value="Submit">
</form>
```

When this form has been submitted you should receive what appears to be a blank page.

Please use your web browsers “view source” command to examine the content returned.

This should reveal the line to cut-and-paste into your stores “pay now” form. For example

```
<INPUT name=epdqdata type=hidden value="otx7cGHs8od9G3ZAsjO7gw3fjeTJ3O">
```

## Periodic Orders (Recurring Billing)

The CPI provides a facility for performing recurring transactions. This allows you to debit customers on a subscription basis, without them having to re enter their credit card details each time.

In order to use the periodic billing functionality you should be aware of the best practice guidelines. These are:

- You must gain your customers consent to the periodic billing payment. This can be in the form of an email or an 'I agree' option on your web site, which should be recorded and provided in the event of a dispute.
- You must make customers aware of the terms of the periodic billing payment (i.e. the amount and frequency).
- You must display clear cancellation instructions on your web site.
- Periodic Billing transactions must not be submitted for Maestro or Solo cards.

The CPI Periodic Billing function allows you to send the amount, frequency interval (i.e. once, twice, 100) and the frequency cycle (i.e. daily, weekly, monthly) to the ePDQ processing engine. ePDQ will then automatically process the transaction.

For further information on the instructions for integrating recurring billing into your ePDQ solution, please refer to the ePDQ CPI Integration Extensions document, available from <http://www.epdq.co.uk/nextsteps/cpi.htm>.

Currently, if you use Internet Authentication, only the first transaction is covered. You may be charged back for subsequent periodic billing transactions.

Please note that Periodic billing orders are not permitted on Solo or Maestro cards

## Handling the returned response

### Step 4 and 5

After ePDQ has processed an authentication and authorisation request, the outcome of the transaction can be sent to the POSTURL so that you can, for example, automate the fulfilment process. The post to your server is a background task that is totally independent of the confirmation page. If you wish to show your own confirmation page, your storefront may need to retrieve data from an updated database in order for the page to display all of the necessary detail.

After this post has taken place, the order confirmation screen is presented to the purchaser by ePDQ. Thus, with ePDQ you are not reliant on the consumer pressing the “continue” button on the order confirmation screen in order to receive the transaction details. When the customer clicks on continue, they will be re-directed to the RETURN URL destination.

Of course, if your storefront does not need to know the outcome of the transaction, you can turn off the Post Order Result function using the ePDQ Configuration Service described earlier.

The response returned is only the authorisation response (i.e. Success or Declined). The authentication response will be sent as a separate value and should not be provided to the cardholder (this will help to eliminate the opportunity for fraud). Typically the card issuer will advise their cardholder if they have passed or failed authentication.

## Order Tracking

In Steps 4 and 5 your store will receive the outcome of the transaction. When your store receives information from the CPI, it is important to know which order this applied to. The two methods of transaction confirmation (PostURL to handle ePDQ status response and ReturnURL to return the cardholder to your store) require different methods of validation.

**PostURL.** The transaction response variables are posted back automatically to this URL. Here, you would typically invoke an order fulfilment script to populate your ‘successful’ database, send email confirmation, organise orders etc. and basically update your database for that particular transaction.

As in any automated online environment we would recommend the inclusion of some kind of “exception handling” in case of a failure to receive the transaction status data (communication problems, server unavailability etc.)

If you do not create your own order id references then you will need to access the ePDQ Store ([https://secure2.epdq.co.uk/cgi-bin/ClearCommerce\\_Engine/](https://secure2.epdq.co.uk/cgi-bin/ClearCommerce_Engine/) client id) in order to manage transactions.

**ReturnURL.** The cardholder is returned to this URL address after clicking ‘Continue’ on the ePDQ CPI response page. In order to assist with identification, and allow you to reunite the customer with the transaction status data sent to the Post URL, the order id is also sent to the return URL using a GET method.

Note that using the IP address for identification is not recommended since it is possible for a number of people to be ‘sharing’ an IP address, for example a large organisation.

## Program Requirements (Response Handling Program Invoked by the POST URL)

For added security, your response handling program must be username / password protected using basic server authentication. You may find it easier to place the response handling program in its own directory and apply username / password protection to the entire directory.

Your Systems Administrator can configure this for Windows based servers. For Unix / Linux based servers you should refer to your hosting company/ISP.

You will need to ensure the script handling the transaction status POST returns a valid HTML page. If the script handling the post returns nothing, the CPI may incorrectly report an error with the process resulting in an 'error' email being sent to the POST email address.

The response handling program will need to be able to accept the following (case sensitive) data parameters from an HTTP POST:

**Please note:** The return URL value defines the location on your server that the customer should return to once the transaction is complete. The oid is sent via GET method attached to the return URL to allow you to identify a returning customer.

The Post URL defines the location on your server that the transaction status is posted to. This is a standard web post, protected by basic authentication, and the Post URL would typically be a script which updates your database with the transaction status information.

If your transaction fulfilment system relies upon the transaction status value being successfully received, you *must* ensure your response handling program receives the data successfully. Failure to receive the transaction status (as a result of communication failures, server failure etc.) could lead to customers experiencing difficulties in completing the transaction process. Adequate 'exception handling' should be employed in all internet applications which rely on input from a remote location.

CPI Parameter	Description
transactionstatus	Outcome of the authorisation request. Full values are provided below.
total	Total numeric value presented for authorisation  Range: 1.00 to 999999999.00 Size: Up to 12 characters
Clientid	Numeric client id (also known as your Store ID) assigned by Barclaycard Business . This is different to your merchant id and terminal id.
Oid	Text string containing the order identification number. If a value was not supplied in your originating request, a unique order number will be generated by ePDQ  Size: Up to 36 characters
chargetype	Text string indicating the chargetype specified for the transaction. (Either Auth or PreAuth).
datetime	Text string indicating when the order was submitted by the ePDQ CPI checkout. The date will always be in this format.  e.g. Jan 23 2005 15:45:51
ecistatus	The Internet Authentication response. Only provided if you are enrolled in Internet Authentication and the transaction was Visa or MasterCard.

CPI Parameter	Description
cardprefix	First digit of the supplied card number. Only provided if you are enrolled in Internet Authentication and the transaction was Visa or MasterCard.

### Transaction Status Results

As shown in the table above, a number of results can be returned. The full (case sensitive) messages returned are detailed below. You may choose to code these into your storefront to provide an appropriate response to your customer when returning to your site.

- a) Success
- b) DECLINED
- c) DECLINED. This store does not accept American Express. Please try again using a different card type.
- d) DECLINED. This store does not accept this card type. Please try again using a different card type.
- e) Awaiting confirmation. Please contact the merchant and quote Ref 2.

This message is generated if the card issuing bank refers the transaction. The cardholder will see a message declining the transaction.

- f) Awaiting confirmation. Please contact the merchant and quote Ref 3.

This message means the issuer has requested a voice authorisation. The cardholder will see the transaction has been declined.

- g) An error has occurred. Please contact the merchant and quote Ref 51.

This typically signifies that the connection timed out and the transaction could not be processed.

- h) There was a Problem processing your order. Please contact the merchant and quote Ref **\*\*ID\*\***.

The CPI will attach an error code to each error message generated. The most common error messages can be found in Appendix D of this guide and these error codes can be matched against the list provided in the main ePDQ user guide.

## Cardholder Transaction Status Message

After sending the post information back to the POST URL the CPI displays a message to the cardholder. The full description is below and consists of a brief description that shown at the top of the Order Status Screen next to a label "Transaction Status" and, for declined transactions; a more detailed description is shown at the bottom of the screen.

a) Success

b) DECLINED

c) DECLINED. (see below)

\*\*This store does not accept American Express. Please try again using a different card type.

d) DECLINED. (see below)

\*\*This store does not accept this card type. Please try again using a different card type.

e) DECLINED.

\*\*Your card was not authorised by your card issuer. As this is an internet transaction it has been declined. Please contact the merchant and quote Ref 2

f) DECLINED

\*\*Your card was not authorised by your card issuer. As this is an internet transaction it has been declined. Please contact the merchant and quote Ref 3

g) Error. (see below)

\*\*An error occurred whilst processing your order. Your card has not been charged. Please contact the merchant and quote Ref 51

h) Error. (see below)

\*\*An error occurred whilst processing your order. Your card has not been charged. Please contact the merchant and quote Ref ID.

## Internet Authentication Transaction Response

The following values will be returned within the 'ecistatus' field:

Value Returned	ECI -Visa	ECI - MasterCard	Description
0	07	0	Transaction type does not support Authentication. Examples of this could be a non-Visa/MasterCard card.
1	06	1	An attempted authentication transaction. Either the issuer or the cardholder are not enrolled in Internet Authentication.
2	05	2	A fully authenticated transaction. The cardholder successfully authenticated themselves.
3	NONE	1	Unsuccessful Internet Authentication transaction. The CPI will decline the transaction for Visa transactions.
4	07	1	A non authenticated transaction. This is a standard eCommerce transaction. Your elected setting for 'Continue Option 1' (for Visa transactions) and 'Continue Option 2' (for MasterCard transactions) flags will determine whether this is processed or declined.
5	06	1	BIN range is not enrolled.
6	06	1	Attempts transaction. Either the issuer or cardholder are not enrolled.

It is important to understand which transactions will be covered by the Internet Authentication liability shift. You should consult our Internet Authentication Procedure Guide for further information.

The 'Value Returned' is the same as the value displayed in the 'Payer Security Level' field. This is found in the transaction detail page of your ePDQ store admin.

## Suggested Test Procedures

Where practical, it is advisable to limit the number of live “test” transactions as these will incur transaction charges. The following suggestions can aid evaluation of your stores ePDQ CPI integration.

Your storefront will need to integrate with the ePDQ CPI as follows:

- Entering ePDQ CPI from an originating page on your web site.
- Optionally accepting transaction status information.
- Returning to a standard web page on your storefront.

Testing entry to the ePDQ CPI can be accomplished by submitting test orders and ensuring that a suitable Payment Page is generated that indicates correct order value.

We recommend that you attempt a test transaction (using the Approved/Declined or Random transaction types) from the Store Administrator Tool **point of sale** function to ensure your ePDQ store has been configured correctly.

For general testing it is normally sufficient to ensure that your form details are reaching the CPI successfully, although you may want to perform some test transactions to confirm that your back end systems are functioning. You can either test these by posting to your script locally from a simple HTML form (although this will not allow for the basic authentication process on the folder containing your fulfilment script to be tested), or use the test details below, which will always return a declined response:

Card Number: 4111111111111111 (not suitable for Internet Authentication test)  
Expiry Date: 12/09  
Amount: keep as low as possible

You must ensure that after testing, all transactions are being processed correctly. Any live transaction processed through the Point of Sale tool must be processed in ‘**Production**’ mode.

### Testing for approved responses

Full end to end testing for a successful response using a test card number is not available and can only be done in a live environment by performing transactions on a valid, ‘live’ card. We do not recommend you attempt this as the tests described above should be sufficient. However, if you choose to perform full end to end testing with your own card details, please ensure you use the value **£11.11**. This will allow us to identify these live transactions as tests.

**Note: the card will be charged £11.11 for this transaction therefore please ensure that all test sales are voided on the same day to avoid being charged processing costs.**

## Internet Authentication Testing

The ePDQ CPI has been fully tested by us to comply with the Verified by Visa for Visa cards and SecureCode for MasterCard and Maestro standards. Neither Visa, MasterCard or Maestro allow test card numbers to be supplied to merchants for the purposes of testing Internet Authentication in a live environment.

If you have activated the ePDQ CPI for Internet Authentication, the Verified by Visa logo and SecureCode logo will appear on your payment page. You will see a new result of “ecistatus” returned with each transaction. Regardless of whether authentication was performed or not, this value will be returned. If you cannot see the logos displayed or the ecistatus value on your tests, and believe authentication should be available, please contact us.

### Testing Authentication Transactions

If you are using the Authentication service then we recommend that you attempt a test transaction to ensure that the ‘in-line’ window appears correctly

Card Number: 4012000000009017 (only suitable for Internet Authentication test)  
Expiry Date: 01/09  
Amount: keep as low as possible

## Transaction Reconciliation

As with all transactions, you must ensure that you are able to reconcile the transaction details of each order. If you choose not to pass your own order ID through ePDQ, you will hinder your ability to track and reconcile transactions.

ePDQ offers you three ways with which you can ensure that orders received via the ePDQ CPI match the order ID and transaction price as supplied by your storefront:

- View the details at the ePDQ Merchant Administration Service (Your store admin tool)
- Data returned to your storefront and
- The email (digital receipt) confirming order details and transaction status

Should you have any reconciliation problems with any particular order you should not despatch the goods. If you are distributing soft goods, you should ensure that you employ a real time method of checking order details by reconciling information passed back to your store.

## Putting Your ePDQ Store “Live”

Techniques described indicate how you can integrate and test your storefront with a “live” ePDQ account with minimal transaction charges.

When you are ready to put your store “Live” please inform us to activate your account. You should not need to make any further changes to your ePDQ CPI interface in order to put your store live.

To go live please fill out the ePDQ Account Activation form on the ePDQ CPI next steps web page as provided in the Barclaycard Business welcome letter. This form will require your client/store ID. Please note that we cannot accept telephone requests for security and audit reasons.

Once your account has been activated you will be notified by return email within 24 hours of receipt of your email. Please be aware that upon activation, all authorised transactions passed through ePDQ will be debited from the cardholder’s available spend, and any transactions previously processed on a valid card may well be settled, so ensure you void any test transactions outstanding in the ePDQ database.

**IMPORTANT.** If you perform “test” transactions when your store is live, and refund so that the balance is zero you will still be **charged a processing fee for the authorised transactions**, and **charged for the refund** (please refer to the Barclaycard Business Merchant Agreement, and Terms and Conditions.).

All live transactions from the Store Admin Point of Sale must be submitted in **Processing Mode of Production**. If the transactions are not submitted in Production, you will not receive payment for these transactions. We are unable to re-process these transactions.

## Trouble Shooting

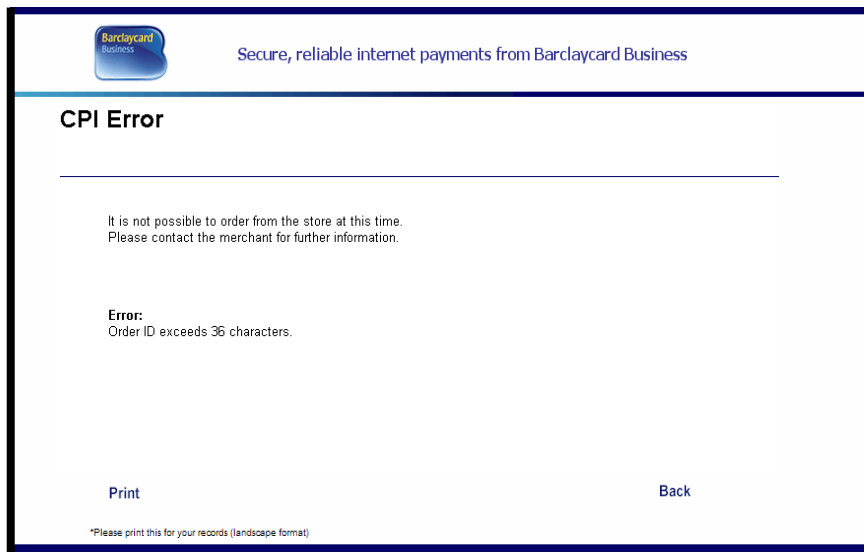
This section is designed to provide you with suggestions and answers to questions that we regularly receive from our customers. Most of the trouble shooting guidelines are based around simple implementation of the product and should allow you to resolve any queries you may have.

If you have a query that is not answered by the information below or wish to discuss a problem in more detail, please contact our eCommerce Support Team.

The details below form the most common problems encountered when integrating the CPI. Typically, the errors occur for a valid reason, which is easily resolved. Please contact us if the solutions below do not solve the problem you are experiencing.

### Error Messages

For any error generated by the CPI, a response will be provided according to the error that has occurred. This will be displayed to the cardholder in a similar way to the example shown below. Should an error occur, you should ask the cardholder to quote the reported error message.



If you experience an error whilst using the CPI payment page, please check the following.

- Ensure that all mandatory data is being sent (epdqdata, returnurl and merchantdisplayname) from your site, and that the fields are formatted correctly (field size and content).
- That the Allowed URL you have set in the CPI Admin tool matches *exactly* the URL from which you are submitting the variables to ePDQ. Any appended session data (e.g. /epdq.cgi?oid=1234 ) will result in a failure. Please also ensure the URL is correctly formed ( e.g. are you submitting from <http://mysite.com> as opposed to <http://www.mysite.com> )
- Ensure that your encryption process has worked correctly, and that you are submitting a valid epdqdata string and not an error message. You should check the source code of the page submitted to ePDQ to ensure you see a correctly formatted tag, e.g.:  
<input type="hidden" name="epdqdata" value="0xctja-ehgwerkqe-234j2liefhewoifhoe87r53">

The CPI will attach an error code to each error message generated. The error code can be matched against the list provided in the ePDQ Operating and Administration guide.

Each error will provide high level descriptive text, as well as the code, e.g.:

“There was a problem processing your order. Please contact the merchant and quote ‘Ref 502’.”

Error code 502 relates to an order that has been marked for review by one of the ePDQ Fraud Rules. You should log into your store administrator tool and identify which rule has been triggered so that you may provide a suitable response to your customer.

### **Declined Transactions**

If you have activated your store and are experiencing declined responses on cards, which you presume to be valid, or the responses are not showing in the ePDQ Store Admin Interface, please check the following.

- The password you are using for the encryption process matches *exactly* the password you have set in the CPI admin tool, and matches a valid user account set on the ePDQ Store Admin. This account must also have full permissions.
- The order id you pass across must be unique, for every order. Failure to comply will result in automatic declines.
- One of the fields keyed in or submitted may exceed the maximum length allowed.

### **PostURL Problems**

If you have processed a transaction, and the transaction status data has not been received by your Post URL fulfilment script, please check the following.

- Check the Post URL is a standard non-SSL page (<http://>).
- Check your server logs first. Ensure that your script is actually being called, and look for any relevant error messages.
- Ensure that you have correctly configured the Basic Authentication on the folder or directory containing your fulfilment script. Failure to activate authentication will result in transaction status data not being processed.
- Check that the Post URL is configured correctly in the CPI Admin tool, and that the username and password you have set matches the configuration on your server.
- Test your script by posting to it locally to ensure the script contains no logic errors etc.

## Frequently Asked Questions

1. **I cannot reconcile the order ID with the response sent back to me and the admin tool, with the order ID I generated on my website.**

If your website (storefront/shopping basket/checkout) generates an order ID, then you must send it within the initial encryption process to ePDQ. This will then ensure that the order ID you supply is consistent with your website and ePDQ. If you choose not to send the order ID, then ePDQ will automatically generate one for you and will be used as the transaction reference.

If you choose not to send the order ID to ePDQ, then you should not use your ID as a reference as it will not match the one ePDQ generated for you.

See **Mandatory CPI Information** section for more details on sending the order ID for encryption.

2. **I'm successfully calling the ePDQ CPI page but am receiving a 'timeout' message.**

You may not be fully set up at Barclaycard Business to communicate with the ePDQ CPI. Please contact the eCommerce Support Team to confirm set up is complete. You can check correct store configuration by submitting a transaction via the Store Administrator Tool point of sale function.

3. **I am unable to access my store Admin Tool as requested in stage 1 of the 'Logging In' section?**

We may not have activated your store. Contact the eCommerce Support Team.

4. **I'm trying to change my store username and password, but it is not working?**

You must ensure that once you have added the new user details, log out and log back in again as the new user, before you delete the original details set by us. If you do not log out, then you will be in effect trying to delete the user details you are logged in as.

5. **What customisation can I do to the payment page?**

To keep in line with your own branding you can change the background colour, text colour and include either your company name, or if held on a secure server, your company Logo. You should ensure that you use appropriate colours if you do decide to change (see the Integration Enhancements document for further information).

6. **Why does my logo have to be held on a secure server?**

Most browsers will not support a page that displays both non-secure and secure items. As the ePDQ CPI is a secure page, all of its content must be secure. If you present a non-secure logo, the browser would prompt the cardholder to 'view only the non-secure items' at which point; much of the CPI page would not be displayed. Some cardholders may actually cancel the transaction if they do not understand what this error message means.

Some of our Merchant Development Partners offer secure logo hosting for ePDQ customers.

**7. Can I accept Multicurrency transactions?**

Multicurrency processing can be offered subject to agreement. You will need to complete a Multicurrency application and be visited by one of our account managers to discuss your Multicurrency plans. For further information please contact our Multicurrency team on 0870 2430950, quoting your existing Barclaycard Business merchant number.

**8. I cannot get to my administration site at the URL quoted in the documents?**

Please ensure you are accessing the correct URL, which will be: [https://secure2.epdq.co.uk/cgi-bin/ClearCommerce\\_Engine/StoreID](https://secure2.epdq.co.uk/cgi-bin/ClearCommerce_Engine/StoreID). The 'Store ID' is the unique number provided to you via email by us. (Please note there is an underscore '\_' between ClearCommerce and Engine).

**9. It says on your website that ePDQ is straight forward to integrate, why do we need to use a developer?**

As with any Internet based product for card processing, a certain level of skill is required. You must ensure that you have experience with the skill sets required, (HTML, CGI Programming using either Perl or ASP) or that you use a developer or Merchant Development Partner to provide these skills.

**10. How do I find a suitable developer?**

All of the ePDQ Merchant Development Partners are listed on the main ePDQ web site at:- [http://www.barclaycardbusiness.co.uk/information\\_zone/products/epdq\\_preferred\\_partners.html](http://www.barclaycardbusiness.co.uk/information_zone/products/epdq_preferred_partners.html)

**11. My customer has checked with their card issuer who advises that the card was authorised, but ePDQ has declined the transaction. What has caused this?**

The transaction may have been declined because the cardholder did not successfully authenticate themselves using Internet Authentication. You should check the 'authentication' response for the transaction. The card may also have been declined by one of the fraud rules you have activated.

**12. Why have I been charged back for a transaction that had an ECI 7 or 0?**

The ECI 7 or 0 values means that the cardholder did not authenticate themselves, possibly because any part of the Internet Authentication service was not available. The transaction was processed as a standard eCommerce transaction.

If you do not wish to process this type of transaction, you should set the 'Continue VbV or CPI Processing' parameter to NO. This will automatically decline ALL transactions where liability shift is not available.

**13. Where can I find more information on Internet Authentication?**

You will have been provided with a comprehensive Procedure Guide when you applied for ePDQ and Internet Authentication. You should refer to this in the first instance. Our website [www.epdq.co.uk](http://www.epdq.co.uk) then go to the section 'Combating on-line fraud' also features further information.

**14. The Verified by Visa or SecureCode logos do not appear on the CPI?**

The logos will only appear if you have signed up for the Internet Authentication service. You must sign an Internet Authentication agreement before you process this type of transaction. If you believe you have signed up for the service and the logos are not appearing, please contact the eCommerce Support Team.

**15. My customers keep telling me that they never see the Internet Authentication in-line window, is something wrong?**

The cardholders may not be registered for the service, meaning the cardholder will not be requested to login via the in-line window. You may still get liability shift for attempting to authenticate them.

## Contacting the Support Team

By Phone: 0870 60 80 355  
By Email: [info.epdq@barclaycard.co.uk](mailto:info.epdq@barclaycard.co.uk)

By Post: eCommerce Support  
Dept. CSD  
1234 Pavilion Drive  
Northampton  
NN4 7SG

Monday to Friday: 8.00am to 8.00pm  
Saturdays: 9.00am to 5.00pm

If you are calling from outside the UK, please dial +44 (0) 1604 254222

To enable us to assist you, it would be useful for you to provide us with as much detail as possible regarding your query. Any supporting information would help, particularly:

- Your ePDQ Store ID.
- Your Internet merchant number (this can be found on your monthly statement).
- Browser name/version
- Name and version of the operating system you are running
- Name and version of your web server
- Name and version of any tools/utilities you are using to install the ePDQ CPI
- Exact time and date of the problem
- Screen shot or details of any error messages you are seeing
- Copy of your ePDQ CPI source code

## APPENDICES: SAMPLE SCRIPTS

IMPORTANT – The following scripts are available in downloadable format (.ZIP file) from <http://www.epdq.co.uk/nextsteps/cpi.htm>. The file name is provided in brackets for reference. The username and password are as provided on your ePDQ Welcome Letter.

The downloadable file contains ALL of the following code examples INCLUDING epdqcpim (essential if you are using Perl). Typically the downloadable file on the website and documentation are updated at the same time. It may be prudent to check the downloadable version before commencing integration.

## Appendix A: ASP

### ASPTear Encryption script (asp\_tear\_example.asp)

This can be used on most Windows NT and Windows 2000 systems in order to encrypt the transaction details. This example is based on the 3rd party ASP component AspTear 1.0 (<http://www.alphasierpapapa.com/lisDev/Components/AspTear/>)

Other alternative components are also available.

Barclaycard Business and ePDQ do not recommend any individual product and cannot accept any liability through use or misuse of these third party products.

```
<%
Const Request_POST = 1
Const Request_GET = 2
Set objTear = CreateObject("SOFTWING.ASPtear")
On Error Resume Next
Response.ContentType = "text/html"
Dim strEPDQ

strEPDQ = objTear.Retrieve( _
    "https://secure2.epdq.co.uk/cgi-bin/CcxBarclaysEpdqEncTool.e", Request_POST, _
    "clientid=3&password=pd&chargetype=Auth&currencycode=826&total=1.23", "", "")

If Err.Number <> 0 Then
    Response.Write "<b>"
    If Err.Number >= 400 Then
        Response.Write "Server returned error: " & Err.Number
    Else
        Response.Write "Component/WinInet error: " & Err.Description
    End If
    Response.Write "</b>"
    Response.End
End If
%>

<FORM action="https://secure2.epdq.co.uk/cgi-bin/CcxBarclaysEpdq.e" method="POST">
<%= strEPDQ %>
<INPUT type="hidden" name="returnurl" value="http://www.store.co.uk/">
<INPUT type="hidden" name="merchantdisplayname" value="My Store">
<INPUT TYPE="submit" VALUE="purchase">
</FORM>
```

## Appendix A: ASP (contd.)

### Microsoft XML Parser (ServerXMLHTTP) Encryption script (asp\_xml\_example.asp)

This is the recommended solution for Windows 2000 / ASP storefronts since the XML component is pre-installed on most Windows 2000 servers.

Some ISP's may also prefer this method since the component required is from Microsoft.

Windows NT users can download the Microsoft component from the following URL:

<http://msdn.microsoft.com/xml/general/xmlparser.asp>

Please note that this component is not suitable for Windows 95 / 98 web servers.

```
<%  
,  
' This is the server safe version from MSXML3.  
Dim objXmlHttp  
Set objXmlHttp = Server.CreateObject("Msxml2.ServerXMLHTTP")  
,  
' If you are getting an error...  
,  
' msxml3.dll error '80070005'  
' Access is denied.  
,  
' ...try using the proxycfg.exe tool for a direct connection:  
  
See Microsoft website for full documentation on the proxycfgtool  
  
objXmlHttp.open "POST", "https://secure2.epdq.co.uk/cgi-bin/CcxBarclaysEpdqEncTool.e", False  
  
objXmlHttp.setRequestHeader "Content-Type", "application/x-www-form-urlencoded"  
  
objXmlHttp.send "clientid=3&password=pwd&oid=123&chargetype=Auth&currencycode=826&total=10.48"  
  
Dim strEPDQ  
  
strEPDQ = objXmlHttp.responseText  
  
Set objXmlHttp = Nothing  
  
%>  
  
<FORM action="https://secure2.epdq.co.uk/cgi-bin/CcxBarclaysEpdq.e" method="POST">  
<%= strEPDQ %>  
<INPUT type="hidden" name="returnurl" value="http://www.store.co.uk/">  
<INPUT type="hidden" name="merchantdisplayname" value="My Store">  
<INPUT TYPE="submit" VALUE="purchase">  
</FORM>
```

## Appendix A: ASP (contd.)

### ePDQ returned response handling script (asp\_response\_example.asp)

This example reads the transaction response from ePDQ and writes it to a log file format *dd-mm-yy—hh-mm-ss-oid.log* .

It is intended for demonstration purposes only.

If you are using Internet Authentication it is recommended you add:

```
fname.WriteLine("ECI Status - " & Request.Form("ecistatus"))
fname.WriteLine("Card Prefix - " & Request.Form("cardprefix"))
```

to the script below, prior to:

```
fname.Close.
```

```
<HTML>
<BODY>
<%
```

```
dim fs,fname,path,timestamp
```

```
if Request.ServerVariables("request_method")="POST" then
```

```
    set fs=Server.CreateObject("Scripting.FileSystemObject")

    path="c:\" 'set your logfile directory path here
    timestamp=day(Date) & "-" & month(date) & "-" & year(date)
    timestamp=timestamp & "--" & hour(time) & "-" & minute(time) & "-" & second(time)

    set fname=fs.CreateTextFile(path & timestamp & "-" & Request.form("oid") & ".log",true)

    fname.WriteLine("OrderID - " & Request.Form("oid"))
    fname.WriteLine("Transaction Status - " & Request.Form("transactionstatus"))
    fname.WriteLine("Total - " & Request.Form("total"))
    fname.WriteLine("ClientID - " & Request.Form("clientid"))
    fname.WriteLine("Transaction Time Stamp - " & Request.Form("datetime"))
    fname.Close

    set fname=nothing
    set fs=nothing
```

```
end if
```

```
%>
</BODY>
</HTML>
```

## Appendix B: ASP.net

### ASP.NET Encryption script

This is the recommended solution that can be used on any server that supports .NET 1.0 Framework or above.

Code Behind  
VB

Public Class Example  
Inherits System.Web.UI.Page

```
#Region " Web Form Designer Generated Code "  
    'This call is required by the Web Form Designer.  
    <System.Diagnostics.DebuggerStepThrough(> Private Sub InitializeComponent()  
    End Sub  
  
    Private Sub Page_Init(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles  
MyBase.Init  
        'CODEGEN: This method call is required by the Web Form Designer  
        'Do not modify it using the code editor.  
        InitializeComponent()  
    End Sub  
#End Region  
  
    Private Sub Page_Load(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles  
MyBase.Load  
        'Put user code to initialize the page here  
  
        'The Following Creates the WebClient Object  
        Dim web As New System.Net.WebClient()  
  
        'The Header Content Type is then set  
        web.Headers.Add("Content-Type", "application/x-www-form-urlencoded")  
  
        'PostData is then declared as data type Byte and populated with the post data  
        Dim postData As Byte() =  
System.Text.Encoding.ASCII.GetBytes("clientid=[clientid]&password=[password]&oid=[orderid]&charge  
type=PreAuth&currencycode=826&total=[total]")  
  
        'The Web object is then used to upload the postdata to the Encryption URL and the response is  
stored in the Response variable  
        Dim Response As Byte() = web.UploadData("https://secure2.epdq.co.uk/cgi-  
bin/CcxBarclaysEpdqEncTool.e", "POST", postData)  
  
        'The response from the post is the converted from Type Byte to String and stored in the session  
variable  
        Session("Response") = (System.Text.Encoding.ASCII.GetString(Response))  
  
    End Sub  
End Class
```

ASPX  
VB

```
<%@ Page Language="vb" AutoEventWireup="false" Codebehind="Example.aspx.vb"
Inherits="WebProject1.Example"%>
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.0 Transitional//EN">
<HTML>
  <HEAD>
    <title>Example</title>
    <meta name="GENERATOR" content="Microsoft Visual Studio.NET 7.0">
    <meta name="CODE_LANGUAGE" content="Visual Basic 7.0">
    <meta name="vs_defaultClientScript" content="JavaScript">
    <meta name="vs_targetSchema"
content="http://schemas.microsoft.com/intellisense/ie3-2nav3-0">
  </HEAD>
  <body MS_POSITIONING="FlowLayout">
    'The Session Variable is then output to the FORM and on form submit is posted to
the CPI
    <FORM action="https://secure2.epdq.co.uk/cgi-bin/CcxBarclaysEpdq.e"
method="post">
      <%= session("Response") %>
      <INPUT type="hidden" name="returnurl" value="http://www.store.co.uk/">
      <INPUT type="hidden" name="merchantdisplayname" value="My Store">
      <INPUT TYPE="submit" VALUE="purchase">
    </FORM>
  </body>
</HTML>
```

## Appendix B: ASP.net (contd.)

### ePDQ returned response handling script (asp.net\_response\_example)

Code Behind

Public Class Response

Inherits System.Web.UI.Page

#Region " Web Form Designer Generated Code "

*'This call is required by the Web Form Designer.*

<System.Diagnostics.DebuggerStepThrough(> Private Sub InitializeComponent()

End Sub

Private Sub Page\_Init(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles MyBase.Init

*'CODEGEN: This method call is required by the Web Form Designer*

*'Do not modify it using the code editor.*

InitializeComponent()

End Sub

#End Region

Private Sub Page\_Load(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles MyBase.Load

*'Put user code to initialize the page here*

Dim fs, fname, path, timestamp

If Request.ServerVariables("request\_method") = "POST" Then

fs = Server.CreateObject("Scripting.FileSystemObject")

path = "c:\ " *'set your logfile directory path here*

timestamp = Day(Date.Now) & "-" & Month(Date.Now) & "-" & Year(Date.Now)

timestamp = timestamp & "--" & Hour(Now) & "-" & Minute(Now) & "-" & Second(Now)

fname = fs.CreateTextFile(path & timestamp & "-" & Request.Form("oid") & ".log", True)

fname.WriteLine("OrderID - " & Request.Form("oid"))

fname.WriteLine("Transaction Status - " & Request.Form("transactionstatus"))

fname.WriteLine("Total - " & Request.Form("total"))

fname.WriteLine("ClientID - " & Request.Form("clientid"))

fname.WriteLine("Transaction Time Stamp - " & Request.Form("datetime"))

fname.Close()

fname = Nothing

fs = Nothing

End If

End Sub

End Class

## Appendix C: PHP

### Encryption script (php\_encryption\_example.php)

This PHP code can be used on with any Web Server that is configured to use PHP, in order to encrypt the transaction details. PHP version 4 is required.

```
<?php

#the following function performs a HTTP Post and returns the whole response
function pullpage( $host, $usepath, $postdata = "" ) {

# open socket to filehandle(epdq encryption cgi)
$fp = fsockopen( $host, 80, &$errno, &$errstr, 60 );

#check that the socket has been opened successfully
if( !$fp ) {
    print "$errstr ($errno)<br>\n";
}
else {

    #write the data to the encryption cgi
    fputs( $fp, "POST $usepath HTTP/1.0\n");
    $strlength = strlen( $postdata );
    fputs( $fp, "Content-type: application/x-www-form-urlencoded\n" );
    fputs( $fp, "Content-length: ".$strlength."\n\n" );
    fputs( $fp, $postdata."\n\n" );

    #clear the response data
    $output = "";

    #read the response from the remote cgi
    #while content exists, keep retrieving document in 1K chunks
    while( !feof( $fp ) ) {
        $output .= fgets( $fp, 1024);
    }

    #close the socket connection
    fclose( $fp);
}

#return the response
return $output;
}

#define the remote cgi in readiness to call pullpage function
$server="secure2.epdq.co.uk";
$url="/cgi-bin/CcxBarclaysEpdqEncTool.e";

#the following parameters have been obtained earlier in the merchant's webstore
#clientid, passphrase, oid, currencycode, total
$params="clientid=$clientid";
$params.="&password=$passphrase";
$params.="&oid=$oid";
$params.="&chargetype=Auth";
$params.="&currencycode=$currencycode";
```

```

$params."&total=$total";

#perform the HTTP Post
$response = pullpage( $server,$url,$params );

#split the response into separate lines
$response_lines=explode("\n",$response);

#for each line in the response check for the presence of the string 'epdqdata'
#this line contains the encrypted string
$response_line_count=count($response_lines);
for ($i=0;$i<$response_line_count;$i++){
    if (preg_match('/epdqdata/', $response_lines[$i])){
        $strEPDQ=$response_lines[$i];
    }
}
?>

<FORM action="https://secure2.epdq.co.uk/cgi-bin/CcxBarclaysEpdq.e" method="POST">
<?php print "$strEPDQ"; ?>
<INPUT type="hidden" name="returnurl" value="http://www.store.co.uk/">
<INPUT type="hidden" name="merchantdisplayname" value="My Store">
<INPUT TYPE="submit" VALUE="purchase">
</FORM>

```

**Please Note:**

If you or your developer are using an older version than PHP 4 you may have backslashes included in the ePDQ Encryption String. You will see a error message "Mandatory Information Missing" to remove the backslashes please replace as shown below:

Replace <?php print "\$strEPDQ";?> with<?php print stripslashes("\$strEPDQ"); ?>

## Appendix C: PHP (contd.)

### ePDQ returned response handling script (php\_response\_example.php)

This example reads the transaction response from epdq and writes it to a log file format *dd-mm-yy—hh-mm-ss-oid.log*.

It is intended for demonstration purposes only.

If you are using Internet Authentication it is recommended you add:

```
fwrite($FILE,"ECI Status - $ecistatus\n");  
fwrite($FILE,"Card Prefix - $cardprefix \n");
```

to the script below prior to:  
fclose(\$FILE);

```
<?php  
  
if (!strcmp(getenv("REQUEST_METHOD"),"POST")){  
  
    $path=""; #set your logfile directory path here  
  
    $timestamp=date("d-m-y--H-i-s");  
  
    $FILE=fopen("$path$timestamp-$oid.txt","a");  
  
    fwrite($FILE,"OrderID - $oid\n");  
  
    fwrite($FILE,"Transaction Status - $transactionstatus\n");  
  
    fwrite($FILE,"Total - $total\n");  
  
    fwrite($FILE,"ClientID - $clientid\n");  
  
    fwrite($FILE,"Transaction Time Stamp - $datetime\n");  
  
    fclose($FILE);  
  
    }  
?>
```

## Appendix D: Perl

### Encryption script

**Note:** The following sample script requires Perl V5.005 or later

The following Perl modules need to be installed on your Web Server. They are available at <http://search.cpan.org>.(if you have problems running the script, confirm with your ISP that they are installed.).

URI-1.19  
Digest-MD5-2.30  
HTML-Tagset-3.03  
HTML-Parser-3.26  
Mime Base 64-2.12  
libwww-perl-5.65  
CGI.pm

You also need to extract the module “epdqpci.pm” from <http://www.epdq.co.uk/nextsteps/cpi.htm> and place it in the same directory as your script based on “epdqpci.pl”(below) on your Web Server

Please set the first line to the location of the “perl” interpreter on your server, and edit the remaining lines to suit your store.

#### epdqpci.pl

```
#!/usr/local/bin/perl

#this example would be part of your checkout preparation cgi script
#this code can be used as a basis for integrating a merchant's existing webstore to ePDQ using PERL

use epdqpci;
use CGI;

$request = new epdqpci;
$query = new CGI;

# Your client id assigned by ePDQ
$request->clientid('1234');

# Your passphrase for ePDQ CPI encryption
$request->passphrase('passphrase');

# Auth=immediate or PreAuth=delayed shipment, amend as necessary
$request->chargetype('Auth');

# 3 character currency code assigned to your client id
$request->currencycode('826');

# order id used for order tracking
```

```
# (this example uses timestamp and ip address)
# amend here to set up orderid to your own requirements
$when = time();
$orderid = "$when.$ENV{'REMOTE_ADDR'}";
$request->orderid($orderid);

# Use this only if your web server has to use a proxy
# $request->proxy("http://yourproxy:8080/");

#for the purposes of this example script, the total is hardcoded
#however this value would normally be obtained from your webstore or your website
#my $total = $checkout_total;

my $total="45.00";

# pass in the total and go and get encrypted epdqdata
$epdqdata = $request->getepdqdata($total);

#the following code prints out a sample form ready for submission to the ePDQ cgi
#typically a merchant would print out their checkout page here

print
    $query->header,
    $query->start_html,
    $query->start_form(-action=>'https://secure2.epdq.co.uk/cgi-bin/CcxBarclaysEpdq.e'),
    $epdqdata,
    $query->hidden(-name=>'returnurl',-value=>'http://www.store.co.uk'),
    $query->hidden(-name=>'merchantdisplayname',-value=>'My Store'),
    $query->submit(-value=>'checkout'),
    $query->end_form,
    $query->end_html;
```

## Appendix D: Perl (contd.)

### ePDQ returned response handling script

**Important:** This example stores the received ePDQ data in a local file. It is intended for demonstration purposes only.

If you are using Internet Authentication it is recommended you amend the block in the script to the one below:

```
open ("LOG", ">> results.txt");
print LOG "$Form{'datetime'}\n , $Form{'oid'}\n , $Form{'total'}\n , $Form{'transactionstatus'}
\n", $Form{'ecistatus'} \n", $Form{'cardprefix'} \n";
close (LOG);
```

The Perl language example CGI program is as follows:

```
#!/usr/local/bin/perl

if ($ENV{'REQUEST_METHOD'} eq 'POST') {
    read(STDIN, $buffer, $ENV{'CONTENT_LENGTH'});
    @pairs = split(/&/, $buffer);
}

foreach $pair (@pairs) {
    local($name, $value) = split(/=/, $pair);
    $name =~ tr/+// ;
    $name =~ s/%([a-fA-F0-9][a-fA-F0-9])/pack("C", hex($1))/eg;
    $value =~ tr/+// ;
    $value =~ s/%([a-fA-F0-9][a-fA-F0-9])/pack("C", hex($1))/eg;
    $value =~ s/<!--(.\\n)*-->/g;
    $Form{$name} = $value;
}

open ("LOG", ">> results.txt");
print LOG "$Form{'datetime'}\n , $Form{'oid'}\n , $Form{'total'}\n , $Form{'transactionstatus'}
\n";
close (LOG);

print<<"NO128";
Content-type: text/html

<title>hello</title>
hello
NO128

exit;
```

The flow of this CGI program can be explained as follows:

- The first line must be set to the location of your web servers Perl program. This is specified by your ISP and is typically either “#!/usr/bin/perl” or “#!/usr/local/bin/perl”
- The second block ensures that a CGI POST has been performed and temporarily stores the information received into a “pairs” data array, with the following block “URL decoding” the returned data into the “Form” data array.
- Finally, some of the ePDQ data returned is appended to the “results.txt” file before exiting the program.

## Appendix E: Common Response Codes

You may on occasion receive a call from a cardholder who has had their transaction declined. Wherever possible you should refer to these error codes to provide an informed response to the cardholder.

The error code will be displayed to the merchant in the following format:

**There was a Problem processing your order. Please contact the merchant and quote Ref “xx “**

The reference provided will be one of the numeric codes below.

These are the most common error codes displayed on the CPI and should cover all errors experienced. For a full list of potential error codes, please refer to the Store Administration Guide.

Error Code	Error Reason
3	The issuer has referred the card for voice referral. You will need to contact our Authorisation service to progress this order.
50	The card issuer has declined the card.
51	The connection to the payment engine has timed out.
52	There is a problem connecting to the payment engine.
500	The transaction has been declined as fraudulent. One of the fraud rules you have set to 'reject' has been triggered. This is typically for pre-process rules (i.e. Block Card Number).
501	The transaction was authorised but has been declined by one of the fraud rules you have set to 'reject'. This is typically for post-process rules (i.e. AVS).
502	The transaction was authorised but has been marked for review by one of your fraud rules you have set to 'review'. You will need to access your store admin tool and either accept or reject the transaction.
1053	The card issuer has requested that the card be retained. If you are experiencing this error, please contact us.
1054	Response is not valid. The response from the issuer is not recognised. This error is rare.