



## ePDQ Cardholder Payment Interface (CPI)

Integration Guide

Version 9.0 released March 2009

Software Version: 5.9 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 Payment Acceptance, 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
7.0	March 08	CPI Configuration character lengths
8.0	August 08	CPI Refresh (Phase 1) Changes
9.0	March 09	Merger of integration and enhancements guides

## Contents

Purpose of this Document .....	5
Intended Audience.....	5
Contacting Us.....	5
Documentation.....	5
Glossary .....	6
Part 1 – Introduction and Getting Started .....	7
What is CPI? .....	7
Required Resources .....	7
What is Internet Authentication? .....	8
Registration .....	8
Deactivating Internet Authentication.....	9
Payment Process.....	10
Screenshots of the CPI Pages.....	12
Getting Started .....	13
Password Security Policy .....	13
Creating a Unique User for Your Integration .....	15
Setting the Continue Options for Internet Authentication.....	16
Optional Integration Options .....	18
Pre-population of Address Details.....	18
Mandate Card Security Code .....	18
Removing the Delivery Address.....	18
Periodic Billing.....	18
Merchant Name or Logo .....	18
Supported Card Types .....	18
Secure Postback .....	18
Tax and Shipping .....	18
Things to Keep a Note of.....	19
Activating Your ePDQ Account .....	19
Testing your Integration .....	20
Testing for Approved Responses .....	20
Part 2 – The Integration.....	21
“Quick Start” CPI Integration .....	21
Step 1 - Configuring the CPI Administration Tool.....	25
Step 2 - Encryption of Transaction Details.....	28
Step 3 - Cardholder Redirection .....	30
Pre-population of Address Details.....	30
Common Formatting Problems .....	32
Mandating the Card Security Code .....	33
Removing the Delivery Address.....	33
Periodic Billing.....	34
Periodic Billing Transactions and Internet Authentication.....	35
Merchant Name or Logo .....	36
Supported Card Types .....	36
Secure Postback .....	37
Tax and Shipping .....	37
Fixed Price Encryption .....	38
Step 4 - Handling Transaction Response Messages .....	40
Exception Handling .....	40
Transaction Status Results .....	41
Cardholder Transaction Status Message .....	42
Internet Authentication Transaction Response .....	43
Common Integration Errors .....	44
Not a Valid Allowed URL .....	44
Passphrase Mismatch.....	44
Encrypted Data Not Present.....	45

Mandatory Information Not Present .....	45
No Data Posted to CPI .....	46
Unable to Read Configuration File.....	46
Processing Multi-Currency Transactions .....	47
Appendix A – Example Scripts .....	49
ASP Encryption Script .....	50
ASP Response Handling Script .....	51
.NET Encryption Script.....	52
.NET Response Handling Script .....	54
PHP Encryption Script.....	55
PHP CURL Encryption Script.....	57
PHP Response Handling Script .....	58
Perl Encryption Script .....	59
Perl Response Handling Script.....	61
Appendix B - Common Response Codes .....	62
Appendix C - What is a 'Jump Page'?.....	63

## 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 Cardholder Payment Interface (CPI). It also provides details on how the CPI handles transactions that can be authenticated under the Internet Authentication service.

## Intended Audience

This document is intended for use by:

- Merchants performing their own integrations
- Web developers working on behalf of merchants
- Merchant Development Partners

It is recommended that the merchant responsible for the account reviews this document to ensure appropriate configuration of the account.

A thorough knowledge of HTML and web-based scripting language is required for successful integration of the CPI.

You must ensure that you have the necessary experience with the required skill sets in order to avoid problems with integration. If you do not have these skills, and do not yet have a web developer, please refer to the list of 'Preferred Partners' on the Barclaycard website at:

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

If you do have any queries or questions whilst reading this document then please feel free to contact us, our contact details can be found in the following section.

## Contacting Us

Telephone Number – 0844 822 2099\*  
Email Address – [epdq@barclaycard.co.uk](mailto:epdq@barclaycard.co.uk)

Opening hours - 8:00am to midnight, Monday to Sunday.

*\* Calls may be monitored or recorded to maintain high levels of security and quality of service*

## Documentation

This document and others which can help you with you integration and store setup can be found at the nextsteps website. Please find the link below with the username and password to access the site.

<http://www.epdq.co.uk/nextsteps/cpi.htm>

Username – nextstep  
Password - welcome

## Glossary

Term	Definition
ASP	Active Server Page – Microsoft’s Server side scripting language.
CGI	Common Gateway Interface – a way of interfacing computer programs
CPAN	Comprehensive Perl Archive Network - A large collection of Perl software and documentation
CPI	Cardholder Payment Interface – secure checkout payment page
ePDQ	Internet payment solution from Barclaycard Payment Acceptance
GBP	Great British Pound (Sterling)
GET	An HTTP command which calls data from one script to another
ISO	International Standards Organisation - 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	PHP: Hypertext Pre-processor – 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

## Part 1 – Introduction and Getting Started

To assist you in your integration of the CPI this document has been split into two parts. Part 1 is intended for both the merchant and developer; it introduces the CPI and explains how to pre-configure the ePDQ store before starting integration. Part 2 is intended for the developer who is integrating the CPI into your website and provides a technical overview of the integration, with coding examples.

This document describes the process for successfully integrating the CPI into your transaction process, explaining how to facilitate communication between your website and the CPI, and how to receive and process the authorisation response messages sent back to your website after a transaction has been processed.

### What is CPI?

The Cardholder Payment Interface (CPI) is a securely hosted, online credit and debit card processing service allowing internet based retailers the opportunity to accept payment for goods or services sold via a website. When a cardholder chooses goods or services from your website and proceeds to the checkout, they are directed to the CPI secure payment environment, where they can pay by debit or credit card.

CPI comes with the following features:

- A secure Barclaycard branded payment page for cardholder reassurance
- Real-time Authorisation
- Built-in Internet Authentication (Verified by Visa and SecureCode) for peace of mind for you and your customers
- Allows the pre-population of billing and shipping address details
- Enhanced fraud control - Additional rules you can set to protect your business from fraudulent card transactions

### Required Resources

The following is an example of the resources you may require whilst integrating CPI. As a first step both the merchant and developer should read through this guide to understand the integration process and ensure they have the skills necessary to complete the integration.

- A website for the cardholder to purchase products through- this may include a shopping cart
- A web developer with thorough knowledge of HTML and web based scripting languages (PHP, ASP, Perl, etc.)
- An CPI Store id to integrate to

## What is Internet Authentication?

The following information is intended for review by both the account owner (the merchant) and the developer responsible for integrating the CPI. Internet Authentication offers a level of liability shift against certain charge-backs so it is imperative that the protocol is understood and implemented appropriately.

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

The CPI automatically checks to see if authentication is possible on every transaction. If it is, the CPI will display an embedded 'in-line window', the contents of which are supplied by the cardholder's Card Issuing bank. This page will generally ask the cardholder to authenticate themselves by way of a password, although the in-line window may also be used to enrol the cardholder in the Internet Authentication service during the transaction process.

If the cardholder is able to authenticate themselves, a value is attached to the transaction, which is then submitted as part of the authorisation request. Even if the cardholder or card issuer is not registered for Authentication, ePDQ will attach a value to the transaction to ensure that the card issuer knows you attempted to authenticate the cardholder.

The Internet Authentication service has a number of technical dependencies as it involves all or one of the following at any given time - merchants, card issuers and Visa or MasterCard. There may be occasions when the cardholder is unable to authenticate themselves, or the service is unavailable. If they fail to authenticate then the transaction will be stopped (Visa cards only) as there is no guarantee that the cardholder can be identified. 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 – please refer to the 'Continue Options' for Internet Authentication on page 14. If you continue processing the transaction after an error has occurred, you will lose liability shift

It is recommended that you review the Internet Authentication Procedure Guide to fully understand how Internet Authentication protocols are applied, and the levels of liability shift available in different circumstances for different cards and cardholders. This guide can be found at:-

[http://www.epdq.co.uk/docs/Internet\\_authentication\\_procedure\\_guide\\_V7.0.pdf](http://www.epdq.co.uk/docs/Internet_authentication_procedure_guide_V7.0.pdf)

## Registration

Before you can use the Internet Authentication service, we must register you with Visa (for VbV) and MasterCard (for SecureCode). 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 with both Visa and MasterCard 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 set up with both Visa and MasterCard. 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 CPI, you must contact the support team and pass a security check, we will then deactivate Internet Authentication from your CPI.

You must be aware that you will no longer be covered by the chargeback protection offered by Internet Authentication and may be subject to higher processing charges 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.

Please note, Internet Authentication for Maestro is mandatory – if you choose to remove this facility you would be required to remove Maestro as an accepted card type, and cease to accept this method of payment.

## Payment Process

The CPI Payment Page is a securely hosted internet payment page which can be integrated into any website to facilitate real time transaction processing.

Typically the website into which the CPI will be integrated will have a shopping cart and order database in place already. Many shopping carts have 'plug ins' for CPI, or can be modified to allow the cardholders to be redirected to the CPI payment page, especially if they are open source.

This integration may require your hosting company to install some specific components depending on the language and environment you are running on.

For all ASP integrations on Windows 2000 server and above you will typically need to make use of the XML parser component which is available as standard on the Windows server application. For any Windows NT server users, this component can be obtained from Microsoft.

If you are using PHP you will need to be running version 4 and above, and will need to ensure you are able to use the functions defined in the example scripts to 'grab' the HTML from our server during the encryption process.

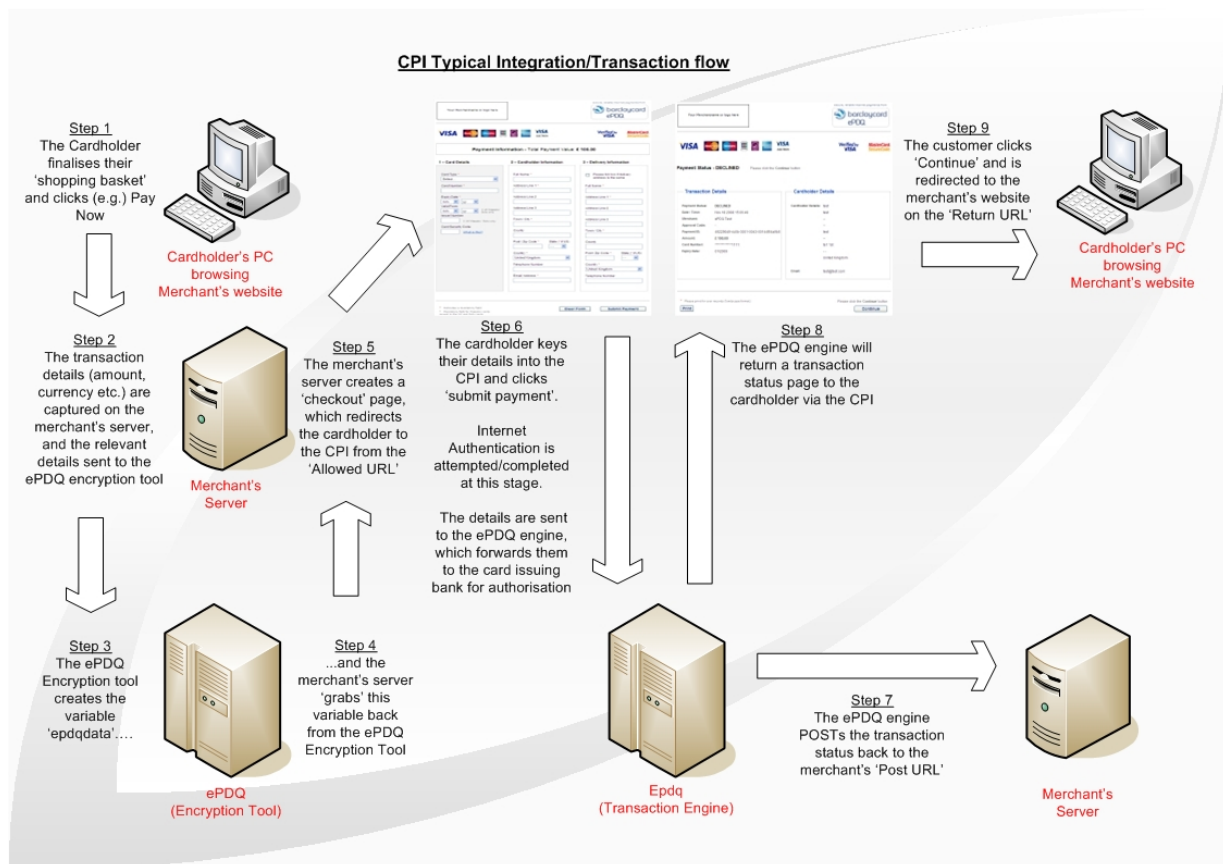
Perl users may need to install certain Comprehensive Perl Archive Network (CPAN) components in order to utilise our encryption script example.

The CPI will:

- Present a payment page to your customers, requesting their credit/debit card details and billing/delivery address
- Submit card information to facilitate Internet Authentication using Verified by Visa for all Visa cards, and SecureCode for MasterCard and Maestro cards
- Submit transactions for real-time authorisation
- Present an order confirmation page to the cardholder, and return transaction status results to a nominated location on your server
- Return the cardholder to your online store

A standard integration would involve the following steps:


- Once the cardholder has decided to buy, your website/shopping cart will redirect them to your final 'Checkout' page
- At the same time, typically, you will send the transaction details (transaction amount, order ID etc.) to the Encryption Tool application on the ePDQ server. The details will be encrypted and generate an HTML form value. We provide example server side scripts for this process in ASP, ASP.NET, PHP and Perl (steps 1 & 2 on the diagram below)
- The cardholder completes their details (name and address) on your Checkout page which you will typically store in the order database
- The cardholder is redirected away from your website over to the secure CPI payment page. This is done using HTML forms (step 3 on the following diagram)
- The transaction is processed on the ePDQ server. The transaction status results (Success, Declined, etc.) are sent to a URL on your server (step 4) using Post method allowing you to update your Order Database and fulfil the transaction accordingly
- The cardholder is redirected to a URL on your server, with their order id attached as session data for identification purposes, allowing you to identify the status of the transaction from your Order Database and display an appropriate message to the cardholder (step 5)









# Screenshots of the CPI Pages



## Standard CPI page

Your Merchantname or logo here

secure, reliable internet payments from  


---

**Payment Information - Total Payment Value: £ 100.00**

<b>1 – Card Details</b> <div style="border: 1px solid #ccc; padding: 5px;"> <p>Card Type * Select... <span style="float: right;">▼</span></p> <p>Card Number * <input style="width: 100%;" type="text"/></p> <p>Expiry Date * mm <span style="margin-left: 20px;">▼</span> yy <span style="margin-left: 20px;">▼</span></p> <p>Valid From mm <span style="margin-left: 20px;">▼</span> yy <span style="margin-left: 20px;">▼</span> † UK Maestro / Solo only</p> <p>Issue Number <input style="width: 100%;" type="text"/> † UK Maestro / Solo only</p> <p>Card Security Code <input style="width: 100%;" type="text"/> <a href="#">What is this?</a></p> </div>	<b>2 – Cardholder Information</b> <div style="border: 1px solid #ccc; padding: 5px;"> <p>Full Name * <input style="width: 100%;" type="text"/></p> <p>Address Line 1 * <input style="width: 100%;" type="text"/></p> <p>Address Line 2 <input style="width: 100%;" type="text"/></p> <p>Address Line 3 <input style="width: 100%;" type="text"/></p> <p>Town / City * <input style="width: 100%;" type="text"/></p> <p>County <input style="width: 100%;" type="text"/></p> <p>Post / Zip Code *    State (* if US) <input style="width: 100%;" type="text"/>    -- <span style="float: right;">▼</span></p> <p>Country * United Kingdom <span style="float: right;">▼</span></p> <p>Telephone Number <input style="width: 100%;" type="text"/></p> <p>Email Address * <input style="width: 100%;" type="text"/></p> </div>	<div style="border: 1px solid #ccc; padding: 5px;"> <p><input type="checkbox"/> Please tick box if delivery address is the same</p> <p>Full Name * <input style="width: 100%;" type="text"/></p> <p>Address Line 1 * <input style="width: 100%;" type="text"/></p> <p>Address Line 2 <input style="width: 100%;" type="text"/></p> <p>Address Line 3 <input style="width: 100%;" type="text"/></p> <p>Town / City * <input style="width: 100%;" type="text"/></p> <p>County <input style="width: 100%;" type="text"/></p> <p>Post / Zip Code *    State (* if US) <input style="width: 100%;" type="text"/>    -- <span style="float: right;">▼</span></p> <p>Country * United Kingdom <span style="float: right;">▼</span></p> <p>Telephone Number <input style="width: 100%;" type="text"/></p> </div>
---	--	--

\* Indicates a mandatory field


† Mandatory field for Maestro cards issued in the UK and Solo cards

Clear Form







Submit Payment



## CPI page with shipping details removed

Your Merchantname or logo here

secure, reliable internet payments from  


---

**Payment Information - Total Payment Value: £ 100.00**

<b>1 – Card Details</b> <div style="border: 1px solid #ccc; padding: 5px;"> <p>Card Type * Select... <span style="float: right;">▼</span></p> <p>Card Number * <input style="width: 100%;" type="text"/></p> <p>Expiry Date * mm <span style="margin-left: 20px;">▼</span> yy <span style="margin-left: 20px;">▼</span></p> <p>Valid From mm <span style="margin-left: 20px;">▼</span> yy <span style="margin-left: 20px;">▼</span> † UK Maestro / Solo only</p> <p>Issue Number <input style="width: 100%;" type="text"/> † UK Maestro / Solo only</p> <p>Card Security Code <input style="width: 100%;" type="text"/> <a href="#">What is this?</a></p> </div>	<b>2 – Cardholder Information</b> <div style="border: 1px solid #ccc; padding: 5px;"> <p>Full Name * <input style="width: 100%;" type="text"/></p> <p>Address Line 1 * <input style="width: 100%;" type="text"/></p> <p>Address Line 2 <input style="width: 100%;" type="text"/></p> <p>Address Line 3 <input style="width: 100%;" type="text"/></p> <p>Town / City * <input style="width: 100%;" type="text"/></p> <p>County <input style="width: 100%;" type="text"/></p> <p>Post / Zip Code *    State (* if US) <input style="width: 100%;" type="text"/>    -- <span style="float: right;">▼</span></p> <p>Country * United Kingdom <span style="float: right;">▼</span></p> <p>Telephone Number <input style="width: 100%;" type="text"/></p> <p>Email Address * <input style="width: 100%;" type="text"/></p> </div>
---	--

\* Indicates a mandatory field

† Mandatory field for Maestro cards issued in the UK and Solo cards

Clear Form

Submit Payment

## Getting Started

Together with the CPI you will also have access to your ePDQ store. This provides you with an interface where you can process manual transactions, issue refunds, control a comprehensive list of fraud rules and manage reports.

We will provide you with a URL, an ePDQ store ID and a user name, once you have received these you will need to contact us on 0844 822 2099 to obtain your password. You are now ready to login to your ePDQ store where you will be presented with the following screen.



The ePDQ store URL will be in the following format:

[https://secure2.epdq.co.uk/cgi-bin/ClearCommerce Engine/\[Your Store ID\]](https://secure2.epdq.co.uk/cgi-bin/ClearCommerce Engine/[Your Store ID])

(Please remember to replace [Your\_Store\_ID] with the store id you have been provided with).

We will now go through some important parts of your ePDQ store which you may wish to configure before you start your integration.

### Password Security Policy

Your ePDQ store is equipped with a password security policy to manage logical access controls which includes password reset and expiration. Within your store you have access to control the policy which, allows the configuration of access controls and password security settings

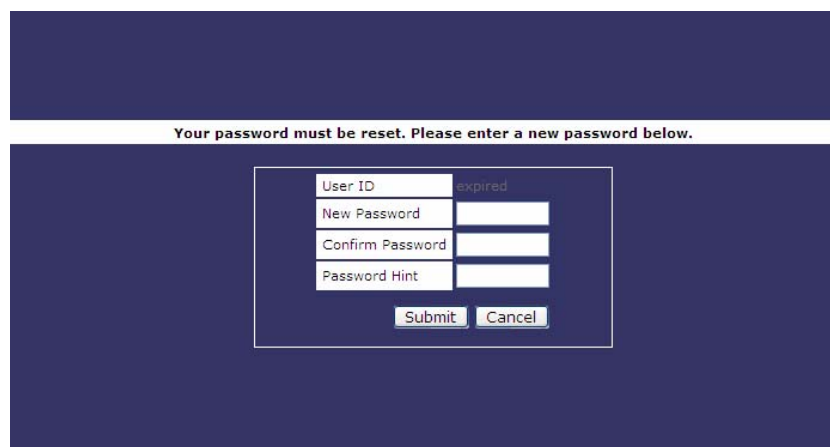
Changing passwords on a regular basis can help tighten security, and the best way to ensure that passwords are changed regularly is to have them expire regularly. ePDQ is equipped with a password security policy so that when a user's password expires, that user cannot sign-on to the ePDQ store until they change the password.

To manage the password security policy, you must first assign yourself with an Administrator who is responsible for managing your ePDQ account together with the configuration of the security policy and management of allocation and editing of user accounts. Typically this can be the initial ePDQ user ID that is allocated to you upon set up.

The options you have available to configure are:

- **User must reset initial password** – This option allows you to specify that users will need to reset their initial password that was supplied when their account was created
- **User must reset password assigned by an administrator** – Similar to the previous if the administrator of your store resets a user's password then the user will be prompted to change their password on the next login
- **Expire passwords after 'X' days** – If enabled this setting can specify when the user passwords will expire and need to be reset
- **Warn user within 'X' days of expiration** – Linked with the previous option this allows you to specify to a user whose password is about to expire how far in advance they are told

The following is an example of the page the user will see when requested to change their password.



Your password must be reset. Please enter a new password below.

User ID	expired
New Password	<input type="text"/>
Confirm Password	<input type="text"/>
Password Hint	<input type="text"/>

To access your security policy settings please follow these steps:

1. Login to your ePDQ store and then select Administration from the menu at the top of the page
2. Then select Security Policy from the left menu, the User and Password Expiration Policy screen is displayed
3. Select the Policy options that are appropriate for your own security policy and then select Apply these changes
4. This policy will then be applied to all users within your ePDQ store

Once the policy has been applied to your ePDQ store you have the flexibility if required, to set an individual user's password not to expire, this may be the user account used for your integration, you can set on the individual user account so the password never expires, this can be achieved by following these steps.

1. Login to your ePDQ store and then select Administration from the menu at the top of the page
2. Select Users from the menu on the left, this will display the User List screen
3. Select the User to change and then select update, the Update User screen is displayed
4. On this screen select the option for Password never expires then select Update, this will apply the change and return you to the User list

More details around the security policy can be found in the 'ePDQ Logical Access Controls – Password Security Policy' document.

[www.epdq.co.uk/docs/protected\\_pdfs/password\\_security\\_policy.pdf](http://www.epdq.co.uk/docs/protected_pdfs/password_security_policy.pdf)

## Creating a Unique User for Your Integration

As part of your initial setup it is recommended that you create a unique login username and password which is used specifically for your integration of the CPI but cannot be used to login to your store. To complete this please follow these steps:-

**NOTE for Developers:** If you are integrating ePDQ on behalf of a Barclaycard customer, please contact the customer to complete this part of the process and to provide you with the unique user ID and password for integration

It is recommended that this unique login is created without the security policy enabled this is to avoid you or your developers the need to reset the initial password. If the security policy is currently enabled within your ePDQ store, prior to creating the unique user you must change your security policy. This can be done by using the following steps.

1. Login to your ePDQ store and then select **Administration** from the menu at the top of the page
2. Then select **Security Policy** from the left menu, the **User and Password Expiration Policy** screen is displayed
3. Make a note of the current Policy options and then **deselect** the **Active** tick box for the options **User must reset initial password** and **User must reset password assigned by an administrator** then select **Apply these changes**

You are now ready to create a unique login user for your integration.

4. Within your ePDQ store select **Administration** on the options along the top
5. Select **Users** from the left side menu, this will display the **User List Page**
6. Select **Add** to create a new user and then configure the following:
  - 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 CPIdev86 is recommended
  - **Password** - must be at least 8 characters long and only supports alpha/numeric characters
  - **Password (confirm)** – Enter the **Password** again for validation purposes.
  - **Password Hint** – Enter a password hint to remind you of the password. This must not be your password and must not compromise the password (i.e. "same as user ID").
  - **Password Never Expires** – This option needs to be **selected** to prevent the password from expiring in the future and stopping your integration from working.
  - **Account Name** – Used for describing the account, example would be '**CPI Integration User**'
  - **Account Description** - Used for describing the account, example would be '**DO NOT DELETE**', flagging to other users not to delete the account
  - **Role** – Set this to '**CPI Access**' this will only allow this account to be used for the integration and not to login to your ePDQ store.
  - **Pagination** – Leave this as default as not required
7. Now select **Add** to create the user, you will then be returned to the **User List Page**

**Remember:** Now that you have created the new user for use with your integration you will need to reinstate the security policy by following steps 1 and 2 above and making the necessary changes to reset your security policy.

**NOTE – Please be aware that this user account will be used for your integration and will be used to submit the transactions. If the password is changed or the user account is deleted then your integration will stop working and you will be unable to process transactions.**

## Setting the Continue Options for Internet Authentication

Before you proceed with the integration of the CPI you need to decide how you want to handle transactions for Internet Authentication.

If a cardholder is registered to use the Internet Authentication service, they will generally 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. The cardholder fails to correctly key in their password after a pre-defined number of attempts.
2. The web page displaying the content provided by the card issuing bank times-out, or fails for any other reason (e.g. cardholder closes page).

Each of these scenarios has an impact on the available level of liability shift. The Continue Options are three 'flags' which control what happens to the transaction in the failure cases.

If the cardholder fails to correctly key in their password for a 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.

You can control the Continue options which can be configured within the CPI Admin Tool for your store. The CPI Admin Tool is accessible via the following link:-

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

**Please note that to change the continue options you will require the ePDQ store clientid and must use the username and password used to complete the integration.**

### Continue Option 1 (Verified by Visa (VbV) - Error Failure)

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

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.

### **Continue Option 2 (SecureCode Error)**

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 may lose the protection afforded by the liability shift by processing these transactions.

### **Continue Option 3 (SecureCode Failed)**

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 will lose the protection afforded by the liability shift by processing these transactions.

For further information, and confirmation of the terms and conditions of Internet Authentication liability shift, please refer to the Internet Authentication Procedure Guide, which can be obtained from the following URL:

[http://www.epdq.co.uk/docs/Internet\\_authentication\\_procedure\\_guide\\_V8.0.pdf](http://www.epdq.co.uk/docs/Internet_authentication_procedure_guide_V8.0.pdf)

## Optional Integration Options

Once you have configured your store and created a user account for your integration you will need to consider the following as part of your integration requirements and advise your developer accordingly. Further details with more technical information can be found further within Part 2 'The Integration' section.

### Pre-population of Address Details

This allows the pre-population of the cardholder address details into the payment pages. By passing across the address values captured on your site will avoid the cardholder having to re-enter the details that they have already entered on your site. In addition it also helps to ensure that the address values supplied to you match the ones supplied for the payment.

### Mandate Card Security Code

This allows you to mandate that the cardholder must enter the Card Security Code value for every payment allowing you to be able to undertake extra security checks on the transaction.

### Removing the Delivery Address

This allows you to choose whether or not the payment page should display the delivery address. This will depend on your requirements for these details, if you do not require a delivery address then you may not wish to capture this information. The default if not specified, is for the delivery address fields to be displayed.

### Periodic Billing

This provides a facility for performing recurring transactions by allowing you to debit customers on a subscription or regular payment basis, without them having to re enter their credit card details each time.

### Merchant Name or Logo

This allows you to specify your company trading name or logo that you want to display on the payment page.

### Supported Card Types

This allows you to indicate which card types your ePDQ store is configured to accept and allows you to control which card logos appear on the payment page.

### Secure Postback

With your integration of the CPI a Postback of the transaction result can be setup, this is sent to a defined Posturl on your site which processes these details, this URL can be either a non-secure (http) or secure (https) depending, on your requirements and the availability of SSL on your hosting server.

### Tax and Shipping

The CPI allows you to separate the transaction total for tax and shipping purposes. You would still submit the Total amount, but you can also submit the delivery costs and tax amount for inclusion within the reporting in your ePDQ Store, and to display on the digital receipts generated by ePDQ.

## Things to Keep a Note of

For future reference and in case you need to make any changes to your integration in the future you should record the following details carefully.

- CPI access username
- CPI access password
- Passphrase
- Post Username
- Post Password

**Important:-** If you have employed a developer to integrate the CPI on your behalf you must ensure that you ask them for these details to avoid any issues when performing future updates..

## Activating Your ePDQ Account

When you have completed your integration and are ready to start processing live transactions you will need to activate your account. The ePDQ account can be activated at any time, although you should be aware that, once active, all transactions authorised will be settled.

The CPI will function (i.e. accept and attempt to process credit cards and post transaction status variables to your Post URL etc.) whether the ePDQ account is active or not.

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

Once we have received your request you should receive notification of the account's activation within 24 hours.

It is recommended that the account owner take responsibility for activation of the account.

## Testing your Integration

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 CPI integration.

Testing entry to the 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 point of sale function within the ePDQ Store interface 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: **4111 1111 1111 1111** (not suitable for Internet Authentication test)  
Expiry Date: **12/11**  
Amount: keep as low as possible

Once you have completed any testing you must ensure that after testing, all 'Live' 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 a low value which you can easily recognise this will allow you to identify these live transactions as tests.

**Note: The card will be charged for these transactions therefore please ensure that all test sales are voided on the same day to avoid being charged processing costs.**

## Part 2 – The Integration

This section contains details to assist with the integration of your site with the CPI; you will require a thorough understanding of HTML and web-based scripting languages to complete this part.

### “Quick Start” CPI Integration

The integration of the CPI is achieved using HTML forms and a small amount of scripting, in conjunction with appropriate configuration of the CPI Administration Tool.

The high level steps for integration are:

#### 1. Configuring the CPI Administration Tool

Access the CPI Administration Tool located at:

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

Here you can configure the following values:

- **Allowed URL** – The address on the merchant’s site where all redirects to the CPI are made from. Typically the final ‘checkout’ page of your shopping cart. This URL should never contain any session data.
- **Post URL** – The URL location on your server where the transaction status results are sent to. The call to this URL is made server to server, so the Post URL script typically is used to update an order database. The Post URL cannot be HTML based for this reason, as there is no browser session utilised.
- **Passphrase** – The unique identifier used by the encryption script to add a security layer to the encryption process. Basically, this value must match the value set for the parameter ‘password’ in the encryption script.
- **Post Username/Password** – ePDQ protects the post-back of transaction status data to your Post URL using Basic Authentication. The values you set here must match the username and password you configure on your web server.

For further information, please refer to **Configuring the CPI Administration Tool**.

#### 2. Encryption of Transaction Details

Before redirecting the cardholder to the CPI payment page, you must encrypt a number of the transaction values. This is done by using a simple script to post the necessary values to an Encryption Tool located on the ePDQ servers. Once encrypted, this tool will generate an HTML form value which will look something like:

```
<INPUT type=hidden name=epdqdata  
value="3F3CCB89A399D038EA641BF7ED75BB8563187C860EF3F1C04A0611F6ABC932E  
AF123E2B6E5A28901FAE105A223A1BBB414A499C815A57F6C699235672960644482912  
CE940C2C770F48FFFF2B8936B6B1CB3547BE5BF0DDC8B00827DEDA08D2C30400AF3  
863A7D983997C0A56186A926">
```

Example scripts in PHP, Perl, ASP.NET and ASP can be found on the ePDQ next steps webpage and in Appendix A of this document.

For further information, please refer to **Encryption of Transaction Details**.

### 3. Cardholder Redirection

Once you have encrypted the transaction details, your website will redirect the cardholder to the CPI Payment page using standard HTML forms.

#### Mandatory Information

The HTML form you use to redirect the cardholder to the CPI payment page must include the following:

- The Action for the Form must be set to the CPI URL:

```
<FORM action="https://secure2.epdq.co.uk/cgi-bin/CcxBarclaysEpdq.e" method=POST>
```

- The epdqdata variable. This represents the encrypted transaction data, as described in the previous step:

```
<INPUT type=hidden name=epdqdata  
value="3F3CCB89A399D038EA641BF7ED75BB8563187C860EF3F1C04A0611F6ABC932E  
AF123E2B6E5A28901FAE105A223A1BBB414A499C815A57F6C699235672960644482912  
CE940C2C770F48FFFF2B8936B6B1CB3547BE5BF0DDC8B00827DEDA08D2C30400AF3  
863A7D983997C0A56186A926">
```

- The returnurl defines the URL on your site where you want the cardholder returned once the transaction has been completed:

```
<INPUT type=hidden name=returnurl value="http://www.your_epdq-  
enabled_website.com/thankyou.script">
```

(The oid value specified in the encryption process will be appended to the returnurl, allowing identification of the cardholder when they return from the CPI to your website.)

- The merchantdisplayname will contain the trading name of the company for display on the CPI Payment Page. This should be easily identifiable by the cardholder and should match the trading name that you have registered with Barclaycard:

```
<input type="hidden" name="merchantdisplayname" value="My Store">
```

#### Optional Information

The HTML form you use to redirect the cardholder to the CPI payment page can also include the following:

- Customisation options for the payment page

```
<INPUT type=hidden name=cpi_logo value="secure_JPEG/GIF/PNG_URL">
```

- The cardholder's billing address information can be passed over to pre-populate the relevant fields on the Billing details page. It is recommended that you pass these values over to avoid your customer having to re-key information if they have already provided this to you.

```
<INPUT type=hidden name=baddr1 value="address line 1">  
<INPUT type=hidden name=baddr2 value="address line 2">  
<INPUT type=hidden name=baddr3 value="address line 3">  
<INPUT type=hidden name=bcity value="City">
```

```
<INPUT type=hidden name=bstate value="US State">
<INPUT type=hidden name=bcountyprovince value="County">
<INPUT type=hidden name=bpostalcode value="Postcode">
<INPUT type=hidden name=bcountry value="GB">
<INPUT type=hidden name=btelephonenumber value="01111 012345">
<INPUT type=hidden name=email value="email@domain.extension">
```

- The delivery address information, if appropriate, can also be passed over to pre-populate the relevant fields on the Shipping details page. It is recommended that you pass these values over, if you offer a separate shipping address, to avoid your customer having to re-key the information.

```
<INPUT type=hidden name=saddr1 value="Address line 1">
<INPUT type=hidden name=saddr2 value="Address line 2">
<INPUT type=hidden name=saddr3 value="Address line 3">
<INPUT type=hidden name=scity value="City">
<INPUT type=hidden name=sstate value="US State">
<INPUT type=hidden name=scountyprovince value="County">
<INPUT type=hidden name=spostalcode value="Postcode">
<INPUT type=hidden name=scountry value="GB">
<INPUT type=hidden name=stelephonenumber value="01111 012345">
```

For further information on redirecting the cardholder to the CPI and for a full list of all of the parameters and values which can be passed over to the CPI, please refer to Cardholder Redirection.

#### 4. Handling Transaction Response Messages

Once a transaction has been processed, the CPI will post the following parameters back to the Post URL:

- transactionstatus – The outcome of the authorisation request.
- oid – The Order ID value.
- total – The transaction amount.
- clientid – Your unique ePDQ identifier.
- chargetype – The type of transaction you requested.
- datetime – Date and time of order.
- ecistatus – The outcome of the Internet Authentication process.
- cardprefix – First digit of the cardholder's card number.

Deriving the final transaction status result is typically the foremost concern of any transaction processing system. The CPI will automatically post back (to your PostURL) a message confirming whether the transaction has been approved or declined. This will occur for all orders processed through the CPI – essentially every time the 'Submit Payment' button is pressed the post back to the Post URL is invoked.

In most typical e-commerce integrations, your Post URL will point to a script which is used to update an order database sitting behind your shopping cart software. The post back at this stage takes place directly from the ePDQ server, and not from the cardholder's browser. This post back takes place after the cardholder has clicked 'Submit Payment' on the CPI, but before the CPI displays the transaction status page.

For further information, please refer to Handling Transaction Response Messages.

## **Returning Cardholder**

Once the cardholder has completed the transaction, and the post back has been made to your Post URL, they will be confronted with the Order Status page on the CPI. On clicking the 'Complete Payment' button, the cardholder will be redirected back to whichever web address you have defined as the Return URL.

The Return URL has the Order ID (oid) parameter appended to it and uses the GET method to redirect the cardholder back to your nominated URL. This allows you to identify the returning cardholder, and typically perform a database lookup on your own server to ascertain what transaction status result was delivered to the Post URL for that particular cardholder.

## Step 1 - Configuring the CPI Administration Tool

The CPI integration process requires that you provide static information regarding the website you are integrating with, e.g. URLs and server side settings. The CPI Admin Tool is used to configure parameters which cannot be defined dynamically on a transaction by transaction basis, such as transaction amount or currency code.

1. Log on to the CPI Administration Tool at:

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

You will need the Client ID (also known as Store ID) and a 'CPI Access' level username and password to log on to the CPI Admin Tool. If you are the ePDQ account administrator, please refer to the section 'Creating a User for Your Integration' if you have not already created an integration user account. If you are a web developer integrating on behalf of a Barclaycard merchant, please contact your client directly to obtain an appropriate username and password.

For security reasons it is recommended that you use only a 'CPI Access' level user account for configuring the CPI Administration Tool.

Please allow up to 15 minutes for these details to be logged by ePDQ once you have submitted them.

2. Configure the following parameters:

Field	Description
Client ID	The Client ID (also known as Store ID) of the ePDQ store you are configuring / integrating. Typically this will already be populated.
Username	The Username used to log into the CPI Admin Tool. Typically this will already be populated.
Password	The Password used to log into the CPI Admin Tool. This will need to be re-entered before the CPI Admin Tool can be updated.
Passphrase	<p>The Passphrase is a value you create which is required for use in the encryption process. The Passphrase must match exactly the value you enter as the password parameter in the script you use for encrypting the transaction details.</p> <p>When the Passphrase is submitted as part of the encryption process (in the parameter 'password'), ePDQ will compare this with the value you have entered in the CPI Admin Tool to ensure they match.</p> <p style="text-align: right;">Max field length - 16 characters Restrictions – Alpha numeric values only</p>

Field	Description
Allowed URL	<p>The Allowed URL is the full web address of the web page you use to redirect your customer from your website to the CPI Payment Page. This must be the full URL including the prefix 'http://' or 'https://' of the page, and must not contain any session data etc.</p> <p>The Allowed URL is a fixed value, so you must ensure that your website always resolves to the same URL before redirecting the cardholder to the CPI.</p> <p>The CPI will check the Allowed URL value for every transaction, comparing the value you have entered into the CPI Admin Tool with the value populating the HTTP_REFERER environment value for every cardholder redirected to the CPI.</p> <p>In some circumstances you may require more than one Allowed URL – for example, you may have multiple domain extensions (.co.uk, .com etc.), or you may have two sites selling the same product. There are a number of possible solutions for resolving this. For example, you could include a 'jump page' to automatically redirect the cardholder to the CPI, or you could use a shared checkout page – i.e. host the checkout page on your .com address and redirect cardholders from your .co.uk address to the .com site.</p> <p>If you are using more than one webpage to redirect cardholders to the CPI, or your site uses session data to transmit information you may need to create a separate 'jump' page to perform the redirect for you. A typical jump page would involve creating a script which generates the necessary HTML, including the required CPI parameters, and automatically posts the data to the CPI. Further information can be found in Appendix E – What is a Jump Page.</p> <p style="text-align: right;">Maximum length 128 characters</p>
Continue Options 1 2 & 3	<p>These parameters control how the CPI handles transactions which may fail the Internet Authentication process.</p> <p>Further information on what these options relate to is provided in the section 'The Continue Options for Internet Authentication'.</p>
Post Order Result	<p>This setting dictates whether ePDQ sends the transaction status results back to your Post URL (defined below). Options are Yes or No.</p> <p>Set this option to 'Yes' if you require the transaction status results posting back to the nominated Post URL on your server.</p> <p>If you set this option to 'No' then you will not receive any post back data notifying the outcome of authorisation requests processed via the CPI.</p> <p>Please note you can elect to have ePDQ send you a conformation email for every transaction processed. To do this contact the eCommerce support team and they will make the change to your ePDQ store.</p>

Field	Description
Post URL	<p>The Post URL needs to contain the full web address of the location on your server where you require the transaction status results sending to after a transaction has been completed, and is only functional if you have set the Post Order Result to Yes.</p> <p>The Post URL can be either a non-secure (http://) or secure (https://) address, further details of the requirements for the secure (https://) postback can be found in the section Secure Postback. Please note the Post URL must not include port numbers.</p> <p>The resource the URL points to must be placed in a Basic Authentication-protected location on your server.</p> <p>ePDQ will retry the post back of transaction status data a maximum of 3 times in the event of any failure to receive the data on the server hosting this URL.</p> <p>The Post URL is 'called' by the CPI after the authorisation of the card has been attempted and processed, but before the CPI displays its own Order Confirmation screen to the cardholder. As such, the Post URL is not called by a browser post – the post back process is completely server side.</p> <p>In a typical e commerce environment you would define the Post URL to point to a script on your server which is used to update an order database attached to the shopping cart, essentially forming part of the transaction fulfilment process, although this will depend entirely on how your online web store operates.</p> <p style="text-align: right;">Max field length – 128 characters</p>
Post Username	<p>The Post Username is the username you have used when configuring 'Basic Authentication' on the directory containing your Post URL script.</p> <p>On a Windows server this is typically achieved via IIS settings. In Unix this is usually achieved by creating a password and .htaccess file.</p> <p style="text-align: right;">Max field length – 16 characters</p>
Post Password	<p>The Post Password is the password associated with your Post Username, and must match the password you have configured on the directory containing your Post URL script.</p> <p style="text-align: right;">Max field length – 16 characters</p>
Post Email	<p>The Post Email is the email address to which the CPI will send a notification email in the event that ePDQ has been unable to successfully post the transaction status to your Post URL after 3 attempts.</p> <p>The email will contain diagnostic information, plus will typically include output from the server which has rejected the CPI's attempt to post the transactions status parameters.</p>

## Step 2 - Encryption of Transaction Details

In order to secure the specific transaction details of any order and try to prevent tampering with parameters such as the transaction amount, the CPI makes use of an encryption process prior to the actual transaction stage.

The encryption process involves collection of the relevant transaction parameters from the cardholder, using whatever tools are provided to you by your online store/shopping cart/order management system, and using a server side script to post these parameters to an encryption program located on the CPI servers hosted by Barclaycard.

The encryption tool takes these parameters and produces some simple HTML form data, which you 'grab' from the encryption tool and use as one of the mandatory form fields in the source code of your website's final 'checkout page'. This form is then used to redirect the cardholder from your website to the CPI payment page.

The Encryption process expects the following values for every transaction:

Parameter	Value
total*	The total amount you wish to charge the cardholder.  Range: 1.00 to 99,999.00
oid	The Order or Invoice number your shopping cart process has generated.  Max Field Length – 36 characters. Must be alpha or numeric. Only special character permitted is '-'
chargetype*	Set to 'Auth' for immediate authorisation and settlement, or 'PreAuth' for immediate authorisation and delayed settlement. These values are case-sensitive
password*	This value should match exactly the Passphrase value configured in the CPI Administration Tool  Max Field Length – 16 characters
currencycode*	The standard ISO code (e.g. 826 for GBP) for the currency you wish the transaction to be processed in. Please refer to Appendix B for details of valid currency code values.  Max Field Length – 3 characters
clientid*	ePDQ administration service Client Id (Also known as your Store ID)  Max Field Length – 20 characters
mandatecsc	This dictates whether the Card Security Code is a mandatory field. 1=on, 2=off  Max Field Length – 1 character

Where \* indicates a mandatory field. If you do not create a value for the Order ID, ePDQ will do this for you. However, we do recommend creation of your own Order ID to assist with order and authorisation response tracking.

The output from the encryption process will be a simple HTML form value, which should look similar to the following:

```
<INPUT name=epdqdata type=hidden value="3F3CCB89A399D038E51B26A0.....">
```

Once the cardholder has chosen to finalise their transaction – i.e. pay via the CPI – you would typically initiate the encryption process. In most cases, this encryption process is a two-stage event:

The steps in this process are:

1. Post the following parameters to the encryption tool using one of our server side script examples, or your own script (the following is from our PHP example):

```
$params="clientid=$clientid";  
$params.="&password=$passphrase";  
$params.="&oid=$oid";  
$params.="&chargetype=Auth"; $params.="&currencycode=$currencycode";  
$params.="&total=$total";
```

2. Once completed, you will receive the 'epdqdata' HTML form value. This needs to be posted to the CPI using a form similar to the very basic example we provide in the script examples:

```
<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>
```

We have included example scripts for ASP, ASP.NET, PHP and Perl, available in Appendix A, 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 perform the encryption process for CPI. Similar techniques can also be adopted with other programming languages. Please see the appendices attached to this document for more information.

The process of encryption is intended for use by websites whose shopping process is dynamic – i.e. the site sells more than one item, or the cardholder is likely to buy more than one item.

### Step 3 - Cardholder Redirection

Once you have performed the encryption process you are ready to redirect the cardholder to the CPI Payment Page, where Barclaycard will collect the cardholder's card details on an SSL enabled web page.

As described previously, there are 3 mandatory fields required by the CPI for all redirects – these are:

Field Name	Description	Max Field Length
merchantdisplayname	The company name which you want to display on the CPI payment page.	25 characters
returnurl	The web address you wish the cardholder to be returned to after the transaction has completed.	100 characters
epdqdata	The data string obtained during the encryption process described previously.	

These form-fields should be present in the HTML form used to redirect the cardholder to the CPI payment page. As well as these mandatory fields you can also post additional fields to the CPI to enhance the cardholder experience. For example, you may need to request some information on your own web site using traditional HTML form elements for the purposes of your order fulfilment process, e.g.:

```
<input type="text" name="email">
```

All fields must be validated prior to submission to the CPI. For example, a badly formatted email address submitted to the CPI will result in an error. The same will occur if the country code you supply is too long, or invalid. Please also ensure all address fields are submitted as alpha/numeric fields only, and do not include any control characters.

The following lists of form values are the only additional values supported by the CPI. You must not submit values created by yourself (for example 'billingname'). These will always be ignored by the CPI and may cause transaction or HTML/JavaScript processing errors.

#### Pre-population of Address Details

To avoid the cardholder the need to re-key information they would have typically already keyed into your website, you can pass across all cardholder information fields, except for their name, to pre-populate the respective fields on the CPI payment pages.

Please note that the cardholder's name is not a supported value due to data privacy.

HTML Form Name	Description	Max Field Length
baddr1*	Text string for the cardholders billing address – line 1	60 characters
baddr2	Text string for the cardholders billing address – line 2	60 characters
baddr3	Text string for the cardholders billing address – line 3	60 characters
bcity*	Text string for the cardholders billing city/town	25 characters
bcountyprovince*	Text string for the cardholders billing county or province (note: US cardholders must complete bstate)	25 characters
bcountry*	Text string for the cardholders billing country code. This can be defined by either the 2 character alpha code (UK) or the 3 digit numeric code (826).  The currency and country code list can be found in the document 'Store Administrator Guide 5.9'. You can also obtain an example of the HTML form used for collecting country codes from the CPI, and from a wide variety of online resources	3 characters
bpostalcode*	Text string for the cardholders billing postal code	9 characters
bstate*	US State code (this is only mandatory where the bcountry is US)	2 characters
btelephonenumber	Numeric telephone number for billing address	30 characters
email*	A string for the cardholder's e-mail address	64 characters
saddr1*	Text string for the cardholders delivery address – line 1	60 characters
saddr2	Text string for the cardholders delivery address – line 2	60 characters
saddr3	Text string for the cardholders delivery address – line 3	60 characters
scity*	Text string for the cardholders delivery city	25 characters
scountyprovince*	Text string for the cardholders delivery county or province (note: US cardholders should complete sstate)	25 characters
scountry*	Text string for the cardholders' delivery country code. This can be defined by either the 2 character alpha code (UK) or the 3 digit numeric code (826).	3 characters
spostalcode*	Text string for the cardholders delivery postal code	9 characters
sstate*	State code (this is only mandatory where the scountry is US)	2 characters
stelephonenumber	Numeric telephone number for delivery address	30 characters

Where \* indicates mandatory fields on the payment page. The details must either be supplied by your storefront or entered manually by the cardholder onto the payment page.

The following is an example of the format these values can be passed in.

```
<input type="hidden" name=" baddr1" value="1 A street">
<input type="hidden" name=" bcity " value="London">
```

## Common Formatting Problems

### Email format checking:

Emails should be formatted cardholder.email@isp.extension

- 'cardholder.email' can be any combination of characters – numbers, letters (upper or lower case) or other symbols.
- 'isp' can be numbers, letters (upper or lower case), underscores or dots.
- '.extension' must start with a dot and there must be letters (upper or lower case) after the dot. There can only be a maximum of 10 letters after the final dot. Common examples are '.com', '.co.uk', '.net' or '.org'.

Emails may also have square brackets after the @

cardholder.email@[255.255.255.0]

Checks **do not** allow:

- More than one @ in the email [ cardholder@email@isp.extension is not allowed]
- .. – two dots in a row [cardholder..email@isp.extension is not allowed]
- .@ -dot just before @ [cardholderemail.@isp.extension is not allowed]
- @. –dot just after @ [cardholderemail@.isp.extension is not allowed]
- @ as the first or last character in the email [cardholder.email@ is not allowed]

In addition there **must** be a dot before the last set of letters for example, [cardholder.email@ispextension is not allowed].

### Country code checking:

We only accept either the ISO numeric 3 digit codes or the 2 character alpha codes. These must be valid codes and of the correct length. No additional characters may be submitted and will cause a transaction to fail.

### Address checking:

The address fields must only be alpha/numeric. They must not include any additional characters, including apostrophes and inverted commas.

It is essential you collect the 'billing address' fields individually. Use of the HTML 'textarea' to collect more than one line of the address may lead to errors on the CPI. These errors are caused by the inclusion of the 'Enter' button and other control characters in any data captured in a textarea as opposed to a single Text Input line.

**Note:** The supply of blank fields (i.e. if the cardholder simply hits the space bar) may cause ePDQ to reject the transaction. The CPI will display the data in the exact format received. This may have implications if you use the Address Verification fraud rules.

## Mandating the Card Security Code

By default the Card Security Code (CSC) field is not mandatory on the payment page. If you wish to ensure that your customers enter their CSC then, this can be achieved by using the variable "mandatecsc" in your integration similar to passing the total and oid values.

To activate the mandate for the CSC specify a value of '1', or a value of '2' to deactivate it. For security purposes this needs to be passed in the string which is encrypted by the encryption tool and returns the epdqdata string. An example for the request string can be found below.

```
clientid=[clientid]&password=[password]&oid=[oid]&chargetype=Auth&total=1.00&currencycode=826&mandatecsc=1
```

## Removing the Delivery Address

The CPI allows you to pass a value to indicate whether or not the payment page displays the "Delivery Information". This option may be used in cases where a delivery address is not appropriate for the product/service concerned (e.g. downloadable products, or where the CPI is collecting payment information for services or products that have already been delivered).

If this option is used, you may still pre-supply the delivery address (as saddr1, saddr2 etc. as described above) however, this address will not then appear on the payment page. If delivery information is pre-supplied, the Payment Status page will display the delivery information otherwise, the cardholder billing address will be displayed.

The variable is sent to the CPI as an HTML hidden field and is defined as follows.

HTML Form Value	Description	Example code
collectdeliveryaddress	<p>This is a value which can turn on and off the "Delivery Information" on the CPI.</p> <p>"0" indicates "do not show the delivery information"</p> <p>"1" indicates "show the delivery information" (CPI normal behaviour)</p> <p>If no value, or any other value is supplied, the CPI defaults to it's normal behaviour where it does display the "Delivery Information"</p>	<p>To remove the Delivery Information from the CPI page:</p> <pre>&lt;INPUT type="HIDDEN" NAME="collectdeliveryaddress" VALUE="0"&gt;</pre> <p>To display the Delivery Information on the CPI page:</p> <pre>&lt;INPUT type="HIDDEN" NAME="collectdeliveryaddress" VALUE="1"&gt;</pre> <p>Any non-zero value will result in the display of the Delivery Information(alternatively, you can choose not to send the "collectdeliveryaddress" parameter)</p>

## Periodic 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. This is achieved by submitting additional data as part of the HTML that is posted to the CPI.

If you wish to perform this type of transaction you must be aware of the best practice guidelines.

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

The parameters sent to the CPI simply define the conditions of the periodic billing transaction, and allow the ePDQ processing engine to repeatedly charge the card in line with the billing requirements specified by you.

Please note you will only receive confirmation of the Transaction Status for the initial transaction as all subsequent payments are handled automatically by the ePDQ engine processes.

HTML Form Name	Description	Example code
orderfrequencycycle	The unit of time for subsequent periodic billing charges:  D – Days (24 hours) W – Weeks (7 Days) M – Months	<code>&lt;input type="hidden" name="orderfrequencycycle" value="value"&gt;</code>  Where value is either D, W or M. (Please note values must be entered as upper case).  If you select M (Month) the transaction will be processed on the same day each month.
orderfrequencyinterval	A number that, when combined with the OrderFrequencyCycle, indicates how often the periodic billing should be run. For example, if OrderFrequencyCycle is set to M and OrderFrequencyInterval is set to 3, periodic billing will take place every 3 months. The default value is 1.	<code>&lt;input type="hidden" name="orderfrequencyinterval" value="value"&gt;</code>  Where value is a numeric indicator.

HTML Form Value	Description	Example code
ordertype	<p>0 - This is a recurring periodic billing order. This is the default value.</p> <p>1 - This is an instalment periodic billing order.</p>	<pre>&lt;input type="hidden" name="ordertype" value="value"&gt;</pre> <p>Where value is either 0 or 1.</p> <p>The following definitions are provided for the purposes of ePDQ Periodic Billing transactions:</p> <ul style="list-style-type: none"> <li>Recurring (value 0) – will charge the cardholder’s account on a periodic basis (i.e. daily, weekly, monthly) for a specified duration or until the order is cancelled, or the full amount is paid. (e.g. an ongoing membership).</li> <li>Instalment (value 1) – will charge the cardholder’s account until the full amount of the purchase has been paid. Typically these will have an end date set (totalnumberpayments). (e.g. a 12 month subscription)</li> </ul>
totalnumberpayments	<p>The number of periodic billing payments. The minimum value is 2, and the maximum is 999.</p>	<pre>&lt;input type="hidden" name="totalnumberpayments" value="value"&gt;</pre> <p>Where value is between 2 and 999. (Use 999 for infinite or extended billing periods)</p>

The CPI will only display the value of the first transaction being authorised. For example, if you submit a recurring order of £10, to be applied 6 times, the CPI will display £10, NOT £60. You must ensure that you provide the cardholder with sufficient information relating to the amount to be charged to their card, and billing arrangements. In addition, the final transaction response page on the CPI will confirm to the cardholder that the transaction is a 'regular payment'.

Full details of how to work with Periodic Billing Transactions can be found in the ePDQ User Guide. This guide is available on-line via the ePDQ next steps web pages, or can be requested by contacting our eCommerce Support Team.

### Periodic Billing Transactions and Internet Authentication

Visa and MasterCard have mandated that only the first periodic billing transaction is covered by the chargeback liability shift. You may still be charged back for the subsequent transactions. Authentication can only be obtained when the cardholder can enter their password at the time of the transaction. As periodic billing transactions are submitted without cardholder intervention this is not possible.

## Merchant Name or Logo

On the payment pages your specified merchantdisplayname value which is passed as part of your integration is displayed alternatively you can display your company logo. This can be done by passing a text string that specifies the location (URL) of a graphical file in GIF or JPG format, in the request sent to ePDQ.

Your logo will need to meet the following dimensions, WIDTH=500 HEIGHT=100. Failure to do this will stretch or shrink your logo. The graphical file must reside on a secure server (with an https:// URL). If this is not practical then you can show your store name instead using the merchantdisplayname form value.

The following is an example of the string which needs to be included in your integration if you wish to display your logo on the payment page.

```
<INPUT type=text name=cpi_logo  
value="https://www.yourepdqenabledwebsite.co.uk/mysecurelogo.jpg">
```

## Supported Card Types

The CPI allows you to pass a value to indicate which card types your ePDQ store is configured to accept and which card logos can be displayed on the payment page.

Please note, the value you pass across to identify accepted card types will not affect which cards you can actually process. The supportedcardtypes variable only relates to the logos that appear on the payment page, and which card types appear in the drop down selection box on the CPI. For example, you may not want to display the Amex logo if you do not have an agreement with Amex cards if you have an agreement with them to process their cards.

HTML Form Name	Description	Example code														
supportedcardtypes	<p>The value describing which card scheme logos appear on the payment page, and which card types appear in the drop down card type selection box.</p> <p>Each card logo has a unique value, as below, all you need to do is select the card logos add the values together and pass that total value across as the supportcardtypes value.</p> <table><tr><td>Electron =</td><td>1</td></tr><tr><td>American Express =</td><td>2</td></tr><tr><td>Solo =</td><td>4</td></tr><tr><td>Maestro =</td><td>8</td></tr><tr><td>JCB =</td><td>16</td></tr><tr><td>MasterCard =</td><td>32</td></tr><tr><td>Visa =</td><td>64</td></tr></table>	Electron =	1	American Express =	2	Solo =	4	Maestro =	8	JCB =	16	MasterCard =	32	Visa =	64	<pre>&lt;input type="hidden" name="supportedcardtypes" value="value"&gt;</pre> <p>For example, if you accept all cards supported by ePDQ then you would submit the value 127 as that is the total of all the card logo values added together.</p> <p>To show all cards except American Express the value would be the total of 1+4+8+16+32+64 which comes to 125, so you would pass the value of 125</p> <p>If you only wanted to show Maestro on the payment page then you would just pass the value 8.</p>
Electron =	1															
American Express =	2															
Solo =	4															
Maestro =	8															
JCB =	16															
MasterCard =	32															
Visa =	64															

## Secure Postback

With the integration of the CPI you can specify either a secure or non secure Postback URL for the transaction results to be posted to. To support the secure Postback you will need to ensure that your secure HTTPS web server is configured to meet the following requirements of security.

- The HTTPS web server must support SSLv3 or TLS
- The web server must support ciphers with encryption keys of 128 bit or more
- The Merchant's web server must have a valid certificate issued for his domain by a recognised certificate issuing authority
- The Certificate must be current (in-date)
- The domain name of the "Post URL" specified through the CPI Admin Tool must match the Common Name (CN) of the Certificate
- Self signed certificates are not allowed
- The Merchant's message receiving script/program under their secure web server must be protected through basic authentication

If these conditions are not met, the CPI will not send the transaction result to your specified POST-back URL.

## Tax and Shipping

The CPI allows you to separate the transaction total for tax and shipping purposes. You would still submit the Total amount, but you can also submit the delivery costs and tax amount for inclusion within the reporting in your ePDQ Store, and to display on the digital receipts generated by ePDQ.

The shipping and tax value will have no bearing on the transaction amount processed by the CPI. It is merely intended for information purposes, and for a more intuitive breakdown of the transaction costs on the digital receipt. However, the total amount of the shipping and tax combined should not exceed the total amount of the transaction.

Failure to comply will lead to error messages on the CPI advising the cardholder there is a contradiction in the transaction amount.

HTML Form Value	Description	Example code
tax	The transaction tax amount to 2 decimal places, e.g. 1.50	<code>&lt;input type="hidden" name="tax" value="value"&gt;</code>
shipping	The transaction shipping/delivery amount to 2 decimal points, e.g. 1.50	<code>&lt;input type="hidden" name="shipping" value="value"&gt;</code>

Note: The tax field relates to State tax and not VAT.

An example is:

```
<INPUT name=epdqdata type=hidden value="otx7cGHs8od9G3ZAsjO7gw3fJeTJ3O">  
(includes the encrypted Total Value of £10)  
<input type="hidden" name="tax" value="2.00">  
<input type="hidden" name="shipping" value="2.00">
```

In this example the total of £10.00 is made up of £6.00 product cost, £2.00 tax and £2.00 shipping. We will authorise the total value of £10.00. We will not add the tax and shipping values to the total.

## Fixed Price Encryption

Some online retail websites may be used for selling only one item, in which case the Encryption process is not required on a transaction-by-transaction basis – i.e. the encryption can be performed only once, and the output received from this can be used for all transactions processed via the CPI. This is referred to as Fixed Price Encryption – you can obtain the relevant HTML epdqdata INPUT tag using some simple HTML instead of implementing a script.

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 epdqdata string you can use the following HTML. Simply copy and paste this into a suitable text editor and save it as, for example, ‘encrypt.html’. Run this HTML file locally, enter the fixed price parameters, and simply view the source code of the page returned after you click ‘Submit’.

```
<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 content should be used in the HTML form used to redirect the cardholder to the CPI – please see the following example which demonstrates the minimum information required by the CPI. All requests to the CPI must include the merchantdisplayname, epdqdata and returnurl parameters:

```
<html>
<head><title>My Store</title></head>
<body>
<form action="https://secure2.epdq.co.uk/cgi-bin/CcxBarclaysEpdq.e" method="post">
<INPUT name=epdqdata type=hidden
value="3F3CCB89A399D038E51B26A044F617AA754F322ADBEC0963A57D71119BCD802
99C82C2D703F13F606666EE91E3F1774A9A0092FAE1D6B9FC8C883EA014CBF17428BC1
686EF053A18B51E95978225CB36">
<input type="hidden" name="returnurl" value="http://www.your-epdq-enabled-
website.com/thankyou.xxx">
<input type="hidden" name="merchantdisplayname" value="My Store">
<input type="submit" value="purchase">
</form>
</body>
</html>
```

## Step 4 - Handling Transaction Response Messages

As soon as the transaction has been processed – i.e. as soon as the cardholder has clicked 'Submit Payment' on the CPI and ePDQ has received the transaction status from the card issuing bank – the CPI will post the transaction status parameters to the Post URL. This will occur for every single transaction processed through the CPI – i.e. every time the 'Submit Payment' button is clicked the CPI will post a set of values back to your nominated Post URL.

The parameters returned to your Post URL can be used to determine the outcome of the transaction and depending on the functionality of your online store instigate the order fulfilment process once the cardholder has returned to your website

In a typical integration, the process would be as follows:

- Cardholder clicks 'Submit Payment' on the CPI
- ePDQ processes the transaction
- CPI 'posts' the transaction status variables to your Post URL
- Cardholder sees transaction response message on the CPI
- Cardholder clicks 'Complete Payment' on the CPI and is redirected to your Return URL

Return URL script identifies returning cardholder through the oid parameter, performs a look-up against your order database using the order id, and displays message appropriate to transaction status.

The oid value is appended to the Return URL as a session variable. When the cardholder redirects to the URL you have supplied you can import this value into an appropriate script hosted on the Return URL to both identify the returning cardholder and display an appropriate message regarding the status of their transaction.

### Exception Handling

It is possible, depending on cardholder behaviour (e.g. clicking the 'Submit Payment' button more than once), for you to receive more than one post back for the same order ID. Please ensure you configure your Post URL script to handle these 'exceptions'. For example, you could configure your Post URL script to create a row in your database for each transaction status posted back, and/or generate an email to inform the merchant of an 'exception' to the 'expected' steps in the transaction process.

It is recommended that you do not over-write the first transaction status message you receive with any subsequent messages you receive for the same order id – ideally you would store all responses for any order, and invoke a process whereby these can be investigated and dealt with appropriately.

## Transaction Status Results

The full list of parameters and values returned to the Post URL are shown below. All the parameters and their values are sent as standard name/value pairs.

Post URL Parameters	Description
transactionstatus	Outcome of the authorisation request.  A list of possible values is shown on the following pages.
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. 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 CPI checkout. The date will always be in this format. e.g. Jan 23 2004 15:45:51
ecistatus	The Internet Authentication response. Only provided if you are enrolled in Internet Authentication and the transaction was Visa, MasterCard or Maestro.  A list of possible values is shown on the following pages.
cardprefix	First digit of the supplied card number. Only provided if you are enrolled in Internet Authentication and the transaction was Visa, MasterCard or Maestro.

The following lists the possible values returned as part of the 'transaction status' parameter, you may choose to code these into your storefront to provide an appropriate response to the cardholder when they return to your site.

- Success
- DECLINED
- DECLINED. This store does not accept American Express. Please try again using a different card type.
- DECLINED. This store does not accept this card type. Please try again using a different card type.
- 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.)

- 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.)
- An error has occurred. Please contact the merchant and quote Ref XX (Please refer to Appendix D for details on the nature of the error code. In the event this message is received the transaction will be declined.)
- 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 error code can be matched against the list provided in the Clear Commerce Software Store Administrator Guide 5.9 documentation. The most common error messages can be found in Appendix D of this 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 is 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' parameter:

ecistatus value	Description – Payer Security Level	What this means
0	Payer Authentication is not supported	No liability shift
1	Payer Authentication supported, cardholder is not enrolled	Attempted liability shift subject to card scheme rules
2	Payer Authentication supported – Authentication successful	Liability shift subject to card scheme rules
3	Payer Authentication supported – Authentication failed	Visa – card will be declined MasterCard/Maestro – no liability shift
4	Payer Authentication supported – Authentication results unavailable  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	No liability shift
5	Payer Authentication supported, card issuing bank is not enrolled	Attempted liability shift subject to card scheme rules
6	Attempted to enrol but the attempt was unsuccessful	Attempted liability shift subject to card scheme rules

The ecistatus parameter can be used to determine the outcome of the Internet Authentication process. In order to understand the extent of liability shift available in each scenario, please refer to the Internet Authentication Procedure Guide:

[http://www.epdq.co.uk/docs/Internet\\_authentication\\_procedure\\_guide\\_V7.0.pdf](http://www.epdq.co.uk/docs/Internet_authentication_procedure_guide_V7.0.pdf)

Further information can also be obtained from the ePDQ User Guide in the section 'Internet Authentication'.

## Common Integration Errors

### Not a Valid Allowed URL

The 'Allowed URL' is the full web address (including http://) of the last web page the cardholder visits before they are transferred to the CPI payment page. It is usually the checkout page and it transfers essential information to the CPI payment page, including details of the cardholder's order.

This error message implies that the specified 'Allowed URL' does not match the URL contained in the HTTP\_REFERER of the cardholder's browser when they are passed to the CPI payment page.

The Allowed URL is the full address (including http://) of the last page of your web site that the cardholder calls at before being transferred to the CPI payment page (this is typically the "checkout" page). This page transfers the information that we need to process the transaction (such as the amount, etc.) to the CPI payment page.

The Allowed URL value is input into the CPI Administration Tool (<https://cpiaadmin.epdq.co.uk/cgi-bin/CcxBarclaysEpdqAdminTool.e>) by your web developer and should match the last web page that the cardholder is redirected from before they arrive on our payment pages.

Typical Reasons for the Allowed URL failing are:

- Omitting the "www" or "http://" at the beginning of the website address
- Ignoring case sensitivity
- Submitting transactions from multiple domains – i.e. your website is registered as both .com and .co.uk, but the Allowed URL is set to the .com address.

### Passphrase Mismatch

Your passphrase is an alphanumeric value created by your web developer when the CPI ePDQ Administration Tool is first configured. ePDQ confirms the identity of transactions submitted to it by using your passphrase rather than your store username or password.

This error message implies that the configured 'passphrase' value differs from the 'password' value supplied during the CPI encryption process.

The passphrase is a value that you create during the integration process, and is used by the CPI to validate your transaction's 'identity' instead of the store username or password.

This value is submitted as part of the encryption script created by the web developer. In the script, the value is referred to as the "password" and in the CPI administration page (<https://cpiaadmin.epdq.co.uk/cgi-bin/CcxBarclaysEpdqAdminTool.e>) it is referred to as the "passphrase".

Please note that this difference is important.

Please note that we do not keep any records of what the passphrase is. The only location where the passphrase can easily be found is in the encryption script itself.

## Encrypted Data Not Present

If an 'Encrypted Data Not Present' message appears, the encrypted transaction data has not been submitted to the CPI.

The encrypted data is posted to the CPI payment page from your website in the HTML form field 'epdqdata'. This information contains values such as total amount, currency code, charge type etc, and is used to create the transaction request on the CPI payment page.

For clarification these fields are:

- The epdqdata field
- The merchantdisplayname field
- The returnurl field

If the HTML input tag "epdqdata" is missing from the FORM used to redirect the cardholder to the CPI then the "Encrypted Data Not Present" error message will appear.

For further information, please refer to Step 2 – Encryption of Transaction Details

## Mandatory Information Not Present

The CPI requires a certain amount of data from your website to be able to process a transaction. This error can occur when some of that data is missing.

This error message implies that one or more of the Mandatory CPI Information fields are missing.

The mandatory information is a set of three fields passed to the CPI by the cardholder's browser when they redirect from your site to the CPI. These 'mandatory CPI information' fields contain the data necessary for us to both process the transaction, and to redirect the cardholder to the appropriate page after payment.

For clarification these fields are:

- The epdqdata field
- The merchantdisplayname field
- The returnurl field

If either the merchantdisplayname or returnurl are missing, or the epdqdata string is present but has any mandatory elements missing (such as the clientid or chargetype) then the cardholder will see the "Mandatory Information Not Present" error.

For further information, please refer to Step 3 – Cardholder Redirection

## No Data Posted to CPI

The CPI requires a certain amount of data from your website to be able to process a transaction. The 'No Data Posted to CPI' error occurs when none of that data is supplied.

This error message implies that all of the mandatory CPI information fields are missing.

For clarification these fields are:

- The epdqdata field
- The merchantdisplayname field
- The returnurl field

If all of the mandatory fields are missing, then the error "No Data Posted to CPI" will show. This error will also appear if your web developer has included an "enctype" attribute in the HTML form tag. The HTML and script examples contained in the CPI Integration Guide show the appropriate format for the Form attribute – no 'enctype' should be specified.

## Unable to Read Configuration File

This error message implies that the CPI Configuration has not been completed, or the clientid specified in the epdqdata string is incorrect.

The CPI configuration file is created by completing the ePDQ Configuration Page. This is typically configured by your Web Developer, and includes values such as the Allowed URL and passphrase.

The "Unable to Read Configuration File" error occurs when:

- The clientid listed in the epdqdata string sent over (for encryption) by your web-server is incorrect
- The ePDQ Configuration Page has not been completed correctly.

In either case the cardholder will see the "Unable to Read Configuration File" error. You should contact your web developer for further assistance.

For further information, please refer to Step 1 – Configuring the CPI Administration Tool.

## Processing Multi-Currency Transactions

The following currencies can be processed via ePDQ:

Currency	Numeric Code	Alpha Code	Decimal Places
Australian Dollar	036	AUD	2
Canadian Dollar	124	CAD	2
China Yuan Renimbi	156	CNY	2
Cyprus Pound	196	CYR	2
Czech Koruna	203	CZK	2
Danish Krone	208	DKK	2
Estonian Kroon	233	EEK	2
Euro	978	EUR	2
Hong Kong Dollar	344	HKD	2
Hungarian Forint	348	HUF	2
Iceland Krona	352	ISK	2
Indian Rupee	356	INR	2
Israel New Shequel	376	ILS	2
Japanese Yen	392	JPY	0
Latvian Lat	428	LVL	2
Lithuanian Litas	440	LTL	2
Maltese Lira	470	MTL	2
Moroccan Dinah	504	MAL	2
New Zealand Dollar	554	NZD	2
Norwegian Krone	578	NOK	2
Polish Zlotych	985	PLN	2
Russian Ruble	643	RUB	2
Singapore Dollar	702	SGD	2
Slovak Koruna	703	SKK	2
South Korean Won	410	KRW	0
Swedish Krona	752	SEK	2
Swiss Francs	756	CHF	2
Sterling	826	GBP	2
US Dollars	840	USD	2
Saudi Riyal	682	SAR	2
South African Rand	710	ZAR	2
Thai Baht	764	THB	2
United Arab Emirates dirham	784	AED	2

In order to process multiple currencies, you must be set up to do so by the Barclaycard Multicurrency Team. We will assign you a specific store and clientid for each currency you accept. By doing this, you will be able to access each store individually for reconciliation purposes. Each store will have a separate clientid, username and password, but can share the same Multicurrency merchant number.

ePDQ will not provide you with exchange rates, so you must ensure that your web site is displaying the appropriate prices at all times.

When submitting a multi-currency transaction to the CPI you must ensure that the correct numeric "currencycode" is submitted during the CPI Encryption process, and that you are correctly set up to do so. Failure to do this will result in the transaction being declined or incorrectly processed. When submitting the numeric currency code ePDQ will look up the relevant currency symbol and display this on the CPI payment page.

The Internet Authentication service is suitable for sterling and Multicurrency transactions, where the card used is a Visa, Maestro or MasterCard card. Liability shift in the event of a charge-back will only be provided if the transaction meets the conditions as set out in the Internet Authentication Procedure Guide.

If you wish to process Multi-currency transactions you should contact the Multicurrency team on 0844 822 2050.

## Appendix A – Example Scripts

IMPORTANT – The following scripts are available in downloadable format (.ZIP file) from <http://www.epdq.co.uk/nextsteps/cpi.htm> or available directly on the following link:

[http://www.epdq.co.uk/docs/protected\\_pdfs/example\\_code2.zip](http://www.epdq.co.uk/docs/protected_pdfs/example_code2.zip)

The downloadable ZIP file contains all of the following code examples.

- [ASP Encryption Script](#)
- [ASP Response Handling Script](#)
- [.NET Encryption Script](#)
- [.NET Response Handling Script](#)
- [PHP Encryption Script](#)
- [PHP CURL Encryption Script](#)
- [PHP Response Handling Script](#)
- [Perl Encryption Script](#)
- [Perl Response Handling Script](#)

## ASP Encryption Script

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.

```
<%
' 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>
```

## ASP Response Handling Script

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.

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

    set fname=nothing
    set fs=nothing

end if

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

## .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=[oid]&orderid=[orderid]&chargetype=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 converted from Type Byte to String and stored in the session variable

Session("Response") = (System.Text.Encoding.ASCII.GetString(Response))

End Sub

End Class

(example.vb)

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>
```

## .NET Response Handling Script

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

## PHP Encryption Script

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>

```

#### 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 an error message "Mandatory Information Missing" to remove the backslashes please replace as shown below:

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

- Please note that according to your PHP setup you may need to adapt the following line and remove the '&' in front of the variables.

```
$fp = fsockopen( $host, 80, &$errno, &$errstr, 60 );
```

## PHP CURL Encryption Script

This script uses the PHP CURL component to extract the epdqdata string from the EncTool.e encryption program.

```
<?php

    $URL = 'https://secure2.epdq.co.uk/cgi-bin/CcxBarclaysEpdqEncTool.e';

    $params =
    "clientid=1&password=password&oid=4&chargetype=Auth&total=100&currencycode=82
    6";

    $user_agent = "Mozilla/4.0 (compatible; MSIE 5.01; Windows NT 5.0)";

    $ch = curl_init();

    curl_setopt($ch, CURLOPT_POST,1);

    curl_setopt($ch, CURLOPT_POSTFIELDS,$params);

    curl_setopt($ch, CURLOPT_URL,$url);

    curl_setopt($ch, CURLOPT_SSL_VERIFYHOST, 2);

    curl_setopt($ch, CURLOPT_USERAGENT, $user_agent);

    curl_setopt($ch, CURLOPT_RETURNTRANSFER,1);

    curl_setopt($ch, CURLOPT_SSL_VERIFYPEER, FALSE); // this line makes it work under
    https

    $result=curl_exec($ch);

    curl_close($ch);

?>

<FORM action="https://secure2.epdq.co.uk/cgi-bin/CcxBarclaysEpdq.e"
method="POST">

<?php print "$result"; ?>

<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>
```

## PHP Response Handling Script

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.

```
<?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");

        fwrite($FILE, "ECI Status - $ecistatus\n");

        fwrite($FILE, "Card Prefix - $cardprefix \n");

        fclose($FILE);

    }
?>
```

## Perl Encryption Script

Note: The following sample script requires Perl V5.005 or later, and according to the version of Perl and the configuration of you hosting server you may need to also include the encypte of 'application/x-www-form-urlencoded' in your form.

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 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;

```

## Perl 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 B - 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 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.

ePDQ Error Code 'xx'	Error Code 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.
CC4	One or more of the parameters you posted to ePDQ is not formatted correctly. Please check for the presence of any control characters or other formatting controls within the parameters you have posted across.
CC6	There has been a timeout with the response from ePDQ. The transaction will have been declined.
CC7	The start date value has been passed over as 'mm/07' for example – i.e. the month part of the start date has not been supplied by the cardholder when entering their card details.
CC8	Indicates a general communication/application failure

## Appendix C - What is a 'Jump Page'?

A Jump Page can be described as a 'stepping stone' between your website(s) and CPI. Typically the jump page itself would be a script to which your website would post all the required CPI variables. This script would then, from a single location, generate a web page 'on the fly' which would automatically redirect the cardholder and post the required HTML form variables to the CPI.

One method for auto-redirection is to include JavaScript commands to automatically submit the HTML form. For example, including the tag `<body onLoad="document.forms[0].submit()">` will automatically submit any form details contained within the same web page.