

SecureTrading API

Xpay4 User Guide

Copyright

© SecureTrading 2011. All rights reserved. No part of this document may be photocopied, reproduced, stored in a retrieval system or transmitted in any form or by any means whether electronic, mechanical or otherwise without the prior written permission of SecureTrading Ltd.

Disclaimer

This document is for informational purposes only. SecureTrading make no warranties, express or implied, through the distribution of this document. No warranty of accuracy is given concerning the contents of the information contained in this publication. To the extent permitted by law no liability (including liability to any person by reason of negligence) will be accepted by SecureTrading, its subsidiaries or employees for any direct or indirect loss or damage caused by omissions from or inaccuracies in this document.

SecureTrading reserves the right to revise the content without obligation to notify any person of such changes.

Document revised on 26-Jan-2011.

Contents

| | | |
|-------|--|----|
| 1 | Introduction | 4 |
| 2 | Setup | 5 |
| 2.1 | Prerequisites | 5 |
| 2.2 | Installation | 6 |
| 2.2.1 | Unpacking | 6 |
| 2.2.2 | Certificate/Key generation..... | 6 |
| 2.2.3 | Certificate Import | 7 |
| 2.2.4 | Security Considerations | 7 |
| 2.2.5 | Configuration..... | 8 |
| 2.2.6 | Port Security..... | 9 |
| 2.2.7 | Running the Xpay4 client | 9 |
| 2.2.8 | Setting up as a service..... | 10 |
| 3 | Your Personalised Program | 10 |
| 3.1 | Start the client | 11 |
| 3.2 | The XML string | 11 |
| 3.3 | Internal Socket..... | 11 |
| 3.3.1 | Using a standard TCP/IP socket..... | 11 |
| 3.3.2 | Using the HTTP POST method | 12 |
| 3.4 | Xpay4 to SecureTrading Gateways and back | 12 |
| 3.5 | Xpay4 result..... | 13 |
| 3.6 | Other Operations..... | 13 |
| 4 | An Example Transaction..... | 14 |
| 4.1 | Communicating with Xpay4 | 14 |
| 4.2 | Result | 14 |
| 5 | XML Specification..... | 15 |
| 5.1.1 | XML Request | 15 |
| 5.1.2 | XML Response | 15 |
| 5.2 | Future Compatibility Considerations | 16 |
| 6 | Troubleshooting..... | 17 |
| 6.1 | Installation problems..... | 17 |
| 6.2 | XML Request problems..... | 18 |
| 6.3 | Contacting support | 18 |
| 6.4 | Xpay4 Characters | 18 |
| 7 | Additional Information..... | 19 |
| 7.1 | Support | 19 |
| 7.2 | Further Reading | 19 |
| 8 | Glossary of terms..... | 20 |
| 8.1 | Authorisation | 20 |
| 8.2 | Good password..... | 20 |
| 8.3 | Payment gateway | 20 |
| 8.4 | Refund..... | 20 |
| 8.5 | Settlement | 20 |
| 8.6 | String..... | 20 |
| 8.7 | XML | 20 |

1 Introduction

Thank you for choosing the SecureTrading API (Xpay4) solution. Xpay4 is a java application that has been created for use by merchants in order to process transactions through SecureTrading (ST2K). Merchants can build a system that is based around a series of XML Requests and Responses which will provide them with greater flexibility in terms of developing their system.

Please note that although the client itself is a java application, you can use any programming language you wish in order to communicate with the client.

The purpose of this document is to provide you with an overview of the installation and configuration of your system and how to begin processing transactions through ST2K. It also aims to provide you with a more detailed understanding of using the Xpay4 system. It details the communications of the Xpay4 system, and also provides detailed examples and explanations of the XML protocol.

2 Setup

2.1 Prerequisites

There are some prerequisites that you will need prior to installing your Xpay4 system; if these prerequisites are not met then Xpay4 may not work.

You will need to ensure that Xpay4 is to be installed on a system where:

- there is a compatible Java environment. This will need to be java 1.6 jdk - To download Java visit the Sun Microsystems Java website (<http://java.sun.com>).
- you have access to a command prompt or shell with administrator access.
- there is Internet access from your server.

The recommended hardware requirements are a 600 MHz processor with 256 MB of RAM. Typical platforms that run the API include IBM compatible computers running a Linux/Windows operating system.

In order for Xpay4 to work, you must ensure that you have a SecureTrading account set up. Once you have this, you will receive a SecureTrading alias which will be supplied to you within your welcome e-mail.

If you do not have your alias but believe you have a SecureTrading account, please contact support (<http://www.securetrading.com/support.html>) with the following information in order to help us find your alias:

- The company name of the merchant.
- The name of the person who signed the original Merchant Agreement.

If you do need to contact SecureTrading support, please be aware that for security reasons we may only speak to an authorized contact of the account.

Please note when using the Xpay4 system in production card details will be taken on your servers, so you will need to ensure that you have your own secure server in place. You may also therefore be required to undertake a form of PCI accreditation, which will differ depending on the size of your company and the volume of transactions that are processed through your system. For more information on PCI accreditation, please visit <https://www.pcisecuritystandards.org>.

Care must be taken to ensure that the Xpay4 and associated files, including any data requested or returned by it, remain secure on the merchant's server.

It is recommended that credit card details are not stored on the merchant's system.

2.2 Installation

For the first part of the installation you will need to ensure that you have the Xpay4 client. In order to obtain this, please contact SecureTrading support, quoting your alias.

The examples outlined below are for MS-DOS and Linux or Unix like platforms, but a similar approach is taken for other operating systems.

2.2.1 Unpacking

You will need to unzip the contents of the xpay4.zip into a new directory on your server, for example `c:\xpay4` on a Windows box or `/usr/local/xpay4` on a Linux or Unix box.

Most Operating Systems now come with a program to allow you to unzip files. If you don't have a copy you can download WinZip from www.winzip.com. Alternatively you could use 7-zip, an open source utility for manipulating archives which can be found at www.7-zip.org.

2.2.2 Certificate/Key generation

Once you have successfully unpacked the contents of the xpay4.zip, you will need to generate your certificate request to SecureTrading's systems, and also your private keystore.

In order to run the example commands below, you will need to ensure that the version of java (1.6) you are using has been included within the "path" environment variable. If java has been saved to the default location, then within the command prompt you will need to type the following:

| Windows Example | Unix or Linux |
|--|--|
| <code>set PATH=%PATH%;C:\Java\jdk1.6.0_10\bin</code> | <code>export PATH=./usr/local/jdk1.6.0_10/bin</code> |

The above commands are **case sensitive**, and assume that Xpay4 has been saved in `c:\xpay4` on a Windows box, or `/usr/local/xpay4` on a Linux or Unix box.

You will need to ensure that keytool is included within your current path. This should have been done when you setup your path above. To check this has been setup correctly, type `keytool` into the command prompt. This should return an output with a list of commands. If you receive an error or nothing at all check that you have setup your path correctly, ensuring that you referenced your directory for the exact version of java that you downloaded.

For the example below, we will use the alias "site1234". You should replace this with your alias, this is your SecureTrading Site Reference. You will need to create your own 'good password' (see glossary) whilst following these commands:

| Windows Example | Unix or Linux |
|---|--|
| <pre>c:\ cd xpay4 java -jar InitXpay4.jar Enter Alias: site1234 Enter new password for site1234.jks: pa55Word</pre> | <pre>cd /usr/local/xpay4 java -jar InitXpay4.jar Enter Alias: site1234 Enter new password for site1234.jks: pa55Word</pre> |

Please note the above commands are **case sensitive** and assume that Xpay4 has been saved in `c:\xpay4` on a Windows box, or `/usr/local/xpay4` on a Linux or Windows box.

Once the above has been run successfully, you should have two new files within your Xpay4 directory:

- **site1234.jks** - This is your private keystore, and must be kept secure as it contains your private key. If the file is compromised you should contact SecureTrading immediately and generate a new keystore in order to ensure that you do not lose any confidential information.
- **site1234.req.pem** - This is your certificate request.

You will need to send your certificate request (site1234.req.pem) to SecureTrading support, who will in return generate a certificate for you.

Please note if you wish to move your XPay4 client to a separate machine, you do not need to go through the process of generating a new keystore, you can transfer the keystore to your new machine.

2.2.3 Certificate Import

Following receipt of your new “.pem” file, SecureTrading support will verify your authenticity, then digitally sign your certificate request and return you two new files. These will be:

- **site1234.cert.pem**, this is your digitally signed certificate from SecureTrading which will be recognized by our systems as matching your account when you attempt to submit a transaction, identifying you as a genuine merchant.
- **merchant_ca.cert.pem**, this is the certification authority used.

You will need to place these files within your Xpay4 directory and then run the following commands:

| Windows Example | Unix or Linux |
|---|---|
| <pre>c: cd xpay4 java -jar ImportXpay4.jar Enter Alias: site1234 Enter keystore password for site1234.jks: pa55Word</pre> | <pre>cd /usr/local/xpay4 java -jar ImportXpay4.jar Enter Alias: site1234 Enter keystore password for site1234.jks: pa55Word</pre> |

This will then attempt to read the files from the current directory and import them into your keystore. You will need to check the output, ensuring it has no errors, and that it has run successfully.

Once this has been completed successfully, you will now have a keystore (site1234.jks) that contains all the necessary certificate chains and authorities. The Xpay4 client itself is within the file “Xpay4.jar”. All other files ending with “.pem” may now be deleted.

You may also change the name of the keystore (site1234.jks) if you wish, but you will need to ensure that you reference it correctly within the configuration file outlined below.

2.2.4 Security Considerations

When performing Xpay4 requests a certificate will be used to establish a secure connection to one of the SecureTrading payment servers. It is important that adequate security measures are adhered to in order to safeguard Xpay4 certificates and prevent certificates from becoming compromised and used by others for illicit use.

When handling Xpay4 certificates it is recommended that these guidelines are followed:

- Physically protect the certificates from unauthorised access
- Restrict access to the certificates to the fewest number of people necessary
- Ensure that expired certificates are deleted and not kept on company systems

If at any time you think that the security of your keystore(s) has been compromised please contact support@securetrading.com immediately.

In addition, you will need to ensure that the keystore and its corresponding password are kept secure from unauthorised users in order to ensure the confidentiality of the encrypted information. You should ensure the keystore file and Xpay4 configuration file are both kept securely and should be accessible only by the Xpay4 client and authorised maintainers.

2.2.5 Configuration

In order for Xpay4 to work, an xml configuration file will need to be setup. By default this file will be called xpay4.ini and will need to be stored within the Xpay4 directory. However you may reference and call the file by another name.

Please find below an example “xpay4.ini” file:

```
<xpay4>
  <port>5000</port>
  <host>127.0.0.1</host>
  <loglevel>INFO</loglevel>
  <printlevel>INFO</printlevel>
  <logfile>xpay4.log</logfile>
  <keystore alias="site1234">
    <file>/path/to/site1234.jks</file>
    <password>pa55Word</password>
  </keystore>
</xpay4>
```

Please find below an explanation of each field (for a breakdown of the codes used please refer to the ST Reference guide):

| Name | Type | Value | Notes |
|--------------------------|------|------------------------------------|--|
| /xpay4/port | O | Int | The port on which Xpay4 will listen for new connections. Default is 5000. |
| /xpay4/host | O | Asc | The host on which Xpay4 will listen for new connections. Default is 127.0.0.1 (localhost) |
| /xpay4/loglevel | O | [ALL, FINE, INFO, SEVERE] | The amount of information Xpay4 will store in the logfile and print to screen respectively. 'ALL' must NOT be used in a production environment since it will cause sensitive information to be logged unencrypted. This can be useful during integration development. 'FINE' provides more information than 'INFO'. 'INFO' is best used for production environment. This is the default value. 'SEVERE' logs a minimal amount of data. |
| /xpay4/printlevel | O | | |
| /xpay4/logfile | O | Asc | URI to a writeable file. Defaults to xpay4.log in the current directory |
| /xpay4/timeout | O | Int | Number of milliseconds to wait for data on idle socket connections before aborting. The default |
| /xpay4/sslconnecttimeout | O | Int | |

| Name | Type | Value | Notes |
|----------------------------|------|-------|---|
| /xpay4/sslreceptivetimeout | O | Int | values should only be changed if you are experiencing problems with connectivity. |
| /xpay4/keystore/@alias | M | Asc | Your username. If Xpay4 is to be used to process transactions through multiple aliases you must provide a separate <keystore> section for each username. |
| /xpay4/keystore/file | M | Asc | URL to the keystore file generated in sections 2.2.2 and 2.2.3. |
| /xpay4/keystore/password | M | Asc | The password for the keystore. |
| /xpay4/retries | O | Int | The number of attempts Xpay4 will try to connect to the SecureTrading gateway network before aborting the transaction. To maintain an uninterrupted service this value should be left at the default value. |

2.2.6 Port Security

Xpay4 uses a number of ports in order to communicate with ST2K and the merchant's system. The ports used by ST2K are outlined below.

| Number | Description |
|--------|---|
| 5000 | This is the default port used to receive incoming requests from your system through an internal socket. Please note that you can change this port to be whichever port you wish by configuring your xpay4.ini file. |
| 443 | This port is used by Xpay4 to send and receive encrypted information to and from ST2K payment gateway network. You will need to ensure that this port is open on your firewall for outgoing connections. |
| 80 | This port is used to check for any new gateways and certificate revocations. Java must also be capable of performing DNS lookup |

2.2.7 Running the Xpay4 client

Once you have successfully completed the stages outlined above, you can begin running Xpay4 with the following commands:

| Windows Example | Unix or Linux |
|---|--|
| <pre>c:\ cd xpay4 java -jar Xpay4.jar</pre> | <pre>cd /usr/local/xpay4 java -jar Xpay4.jar</pre> |

Or, alternatively, if you have changed the name and location of your configuration file, the last command will be:

| Windows or Unix Example |
|--|
| <pre>java -jar Xpay4.jar /path/to/newxpay4.ini</pre> |

Once this has been setup successfully, the command prompt output will look similar to the following:

```
2008-09-11 10:44:47      main    Listening on 127.0.0.1:5000
```

This indicates that Xpay4 is now up and running and is awaiting a connection from your system.

If it has failed to run, please look at the Troubleshooting section of this guide. Alternatively you can contact our support team.

2.2.8 Setting up as a service

Windows

To install Xpay4 as a service on a windows machine you will need administrator access. There is a "InstallXpay4.bat" file located within Xpay4.zip which needs configuring for your system.

| Variable | Notes |
|---------------------|---|
| XPAYDIR | This should be the directory where you installed Xpay4 to. It should be something that is not going to change (if you change/delete this directory the windows service will NOT work). For example, in the case of section " 2.2.1 Unpacking " this variable would be set to <code>c:\xpay4</code> |
| JVMFILE | This should be the location of the file which is within your java directory. It does not matter if this points to the jvm.dll in the client or server sub-directory within the java directory. |
| JAVASERVICESVERSION | You should change this to match the operation system you are using. If it is a 32-bit version you should use "JavaService-32Bit.exe" and "JavaService-64Bit.exe" for 64-Bit OS's. |

Once all these variables have been set you will have to run the "InstallXpay4.bat" file as administrator and follow all on-screen instructions. After this a service will appear in the Windows services list called "Xpay4" enabling you to start/stop Xpay4 as desired.

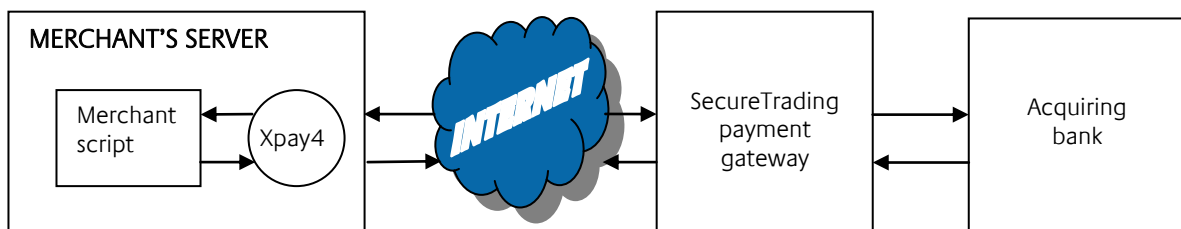
Linux

To install a service on your linux distribution please consult your operating systems manual.

3 Your Personalised Program

The purpose of this section of the document is to provide you with an overview of how Xpay4 works from a merchant's perspective, and also how your program should be setup to process transactions via Xpay4 through ST2K.

Below is a diagram representing the Xpay4 transaction process:



There are five key stages of the process:

- The Xpay4 client needs to be started on the merchant's server
- The merchant's system or script will need to generate an XML String that adheres to the ST XML Specification for each request type.
- The Merchant's system or script will need to send the XML string to the Xpay4 client via an internal socket.

- Xpay4 will then in turn encrypt, send, receive and decrypt the response through the SecureTrading Payment Platform
- Xpay4 will return an XML string back to the merchant's system or script.

Each of these stages is explained in greater detail below.

3.1 Start the client

The first stage before Xpay4 can perform transactions is to ensure it is running and that it is ready for a transaction to be submitted. For instructions on installing the client, please refer to section 2.2 of this document.

3.2 The XML string

The XML string that is submitted to the Xpay4 client will need to adhere to the XML Specification for a given request type, and will include your unique alias and the details of the request you wish to perform.

When submitting a request to SecureTrading, it may be that customer details including their personal information are required. Credit card numbers must be obtained in a **secure** manner, such as the **https** protocol.

The prerequisites section of this document explains the need for PCI accreditation, for this reason SecureTrading strongly recommend that you do not store card details on your server. If you wish to repeat transactions use the unique transaction reference. For more information on the fields required for different requests, please refer to the Child Transactions section of the ST Reference document.

3.3 Internal Socket

Within your application, you will need to open a TCP socket to pass the XML string through to the Xpay4 client so it can process your request. This can be achieved in two ways. Please note that as the XML at this point is only being passed internally from your system to the Xpay4 client, no encryption will be required as the information will not be posted on the internet.

3.3.1 Using a standard TCP/IP socket

This is the recommended method used for programming languages which support TCP sockets such as Java, Perl or Python. The XML string is passed from, by default, localhost (127.0.0.1) and through port 5000. If you wish to change the settings for which port your installation of Xpay4 will be awaiting connections from, please refer to section 2.2.5 Configuration of this document.

As an example, below is pseudo-code which instantiates a socket connection to the Xpay4 client.

```
// Import the socket package
Import Socket

// Define the xml string (see The XML string)
xml = "<RequestBlock><..></..></RequestBlock>\n"

// Instantiate a new socket with IP Address of 127.0.0.1 and Port 5000
s = new Socket ("127.0.0.1",5000)

// Send the xml down the socket
s.send (xml)

// Receive the response from Xpay
Result=s.receive()

// Close the socket
s.close()
// Result is now available for processing by the merchant
```

Following the request being processed by the Xpay4, the data stored in `Result` will be the XML returned.

3.3.2 Using the HTTP POST method

This is recommended for languages which do not support TCP sockets directly, such as Cold Fusion. In this case, the XML String is posted to an internal url.

The equivalent URL that would be entered into an Internet browser would be similar to:

```
http://127.0.0.1:5000/?xml=<RequestBlock><..></..></RequestBlock>
```

In the above example the host address would be 127.0.0.1 and a port of 5000. Inputting the above including the whole XML string that you have generated into a browser is unlikely to be the solution you would prefer. A more suitable way is to program the equivalent in, for example, Cold Fusion:

```
<cfhttp method="post" url=http://127.0.0.1/ port="5000">
<cfhttpparam encoded="No" TYPE="cgi" NAME="xpay4" VALUE="#xml#">
</cfhttp>
```

This assumes that a variable named "xml" has already been created with the relevant XML string as outlined in the XML Specification for the given request type.

The request should not be URL encoded (as indicated by `encoded="No"` in the above example). Your client code may need to specify the **Content-Type** in the HTTP headers to be `text/xml`.

Please ensure a POST is submitted in production as a GET will not be URL decoded and therefore may not work for certain characters.

3.4 Xpay4 to SecureTrading Gateways and back

Once your system has submitted the XML string to the Xpay4 client, then no further action is required until you receive a response. The client will forward the encrypted request to ST2K, which in turn will forward the request onto your acquiring bank. They will then authorise or decline the transaction and then send the response to ST2K which will be encrypted and sent back to the Xpay4 client.

3.5 Xpay4 result

The result is passed to Xpay4 and is then returned down the same internal socket to your script. Once this XML string is available, you can parse the results and build your system around the responses. For example, if your customer was purchasing goods, you could parse out the results of a successful order and then based on these display a success page to the customer and update your records accordingly. The socket must now be closed. Each new transaction requires a new socket connection.

3.6 Other Operations

Other operations performed through ST2K such as refunds, repeats, subscriptions etc all follow the same pattern. Firstly an XML string is generated and sent through a socket to Xpay4. This in turn sends the details to ST2K which will process the request and return a response to Xpay4. Following this Xpay4 will send the response back down the socket to the merchant's script.

4 An Example Transaction

In order to successfully submit a transaction through the Xpay4 systems, you will need to generate and submit an XML string that adheres to the XML Specification for a given request type.

4.1 Communicating with Xpay4

Processing a transaction through Xpay4 is based on sending an XML string through a socket connection to the Xpay4 client.

Included in the xpay4.zip file is an example directory which contains some simple examples of how to write the necessary code to connect to the Xpay4 client and submit a transaction request. There are examples in different languages, but they all follow a similar process. The test examples read in the contents of a “request.xml” file and send it through a socket connection to the Xpay4 client, finally printing the response from the Xpay4 client to the screen.

Also within the example directory is a subdirectory for each request type that can be used with Xpay4. Within each subdirectory is a request.xml and response.xml file which demonstrate how the xml requests and responses will look for each request type.

It is good practice at this point to attempt a test transaction using the example. This will prove that the installation has been successful and that there are no connection issues.

To process a test transaction using the Java (“Test.class”) example:

- Choose a request type to test, for this demonstration we will use the authorisation request.xml (which can be found in the AUTH directory).
- Copy this file to the example directory.
- Open the file, insert your alias (provided when you signed up for the account) within the <certificate> tags and then save the file.
- Open a Command Prompt or Terminal and type the following commands (remember to replace “site1234” with your alias):

| Windows Example | Unix or Linux |
|---|---|
| <code>cd C:\xpay4\example</code> | <code>cd /usr/local/xpay4/example</code> |
| <code>java Test site1234 request.xml</code> | <code>java Test site1234 request.xml</code> |

- Wait for a response to be shown on screen, this should take somewhere between 2 and 20 seconds depending on your connection speed.

4.2 Result

An example of the output you should receive can be found in the example directory.

If you do not receive an output similar to the example, or you do not receive an output at all please refer to the troubleshooting section. If you received an output similar to the one above then you have successfully processed a transaction using the Xpay4 client.

5 XML Specification

This section explains the basic format of the XML request and responses that are used with the Xpay4 system. Full examples of the various requests and responses are covered in detail in the ST XML Specification document.

Please note that the XML Tags outlined within this document are all case sensitive. For more information on XML tags, please refer to <http://www.xml.com>

5.1.1 XML Request

All requests sent to SecureTrading originate from an XML `<RequestBlock>` that has one attribute, `Version`, which refers to the version of Xpay4 you are using. Within the `<RequestBlock>` root element there is currently support for one `<Request>` element which has one attribute, `Type`, which refers to the type of request it is, for example an authorisation or refund request and one `<Certificate>` element which should contain your alias.

An example would be:

```
<?xml version="1.0" encoding="utf-8"?>
<RequestBlock Version="3.51">
.   <Request Type="AUTH" >
.   </Request>
      <Certificate>site1234</Certificate>
</RequestBlock>
```

What is then included within the `<RequestBlock>` tags depends on the type of request that is being submitted to Xpay4, for example an "AUTH" request would be a standard authorisation. The ST XML specification document explains this in full detail.

5.1.2 XML Response

Following a request through the Xpay4 client, a response will be returned similar to the initial request. The response will be within the `<ResponseBlock>` tags.

An example would be:

```
<ResponseBlock Live="FALSE" Version="3.51">
  <Response Type="AUTH">

  </Response>
</ResponseBlock>
```

Within the `<ResponseBlock>` tags, you will receive `<Result>` tags, and within these tags you will be able to determine if a transaction was an error or whether it was successful. If the `<Result>` tag is a 1, then this denotes a successful transaction. The outline of the XML would therefore be similar to the below:

```
<ResponseBlock Live="FALSE" Version="3.51">
  <Response Type="AUTH">
    <OperationResponse>
      <TransactionReference>1-2-2432</TransactionReference>
      <AuthCode>Auth Code:6284</AuthCode>
      <Result>1</Result>
      <Message></Message>
    </OperationResponse>
  </Response>
</ResponseBlock>
```

As with the various XML requests, the responses are outlined in greater detail within the ST XML specification document.

5.2 Future Compatibility Considerations

In order to maintain compatibility with future updates to this specification the following criteria should be adhered to.

- A missing request tag will be considered the same as an empty tag.
- A missing response tag should be considered the same as an empty tag.
- A missing response element should be considered the same as an empty element.
- The order of the tags within a given block may change at any time and should not be depended upon.
- Any unexpected elements should be ignored.
- Any unexpected tags should be ignored.
- Any unexpected attributes should be ignored.
- White-space characters at the start and end of data within a tag should be ignored.

6 Troubleshooting

6.1 Installation problems

If you experience problems installing or starting the Xpay4 application please check:

- you are using the version of Java as specified in Prerequisites.
- Java has been installed correctly, also with the correct PATH set.
- you have exactly one Xpay4 client running on your server.

The following tables show the cause and solution for common installation issues:

| | |
|--|--|
| Exception in thread "main" java.io.IOException: Cannot run program "keytool": CreateProcess error=2, The system cannot find the file specified | |
| Cause: The keytool program cannot be found. This is part of the java installation. Make sure that the path has been set properly (refer to section 2.2.2). | <p>Solution: The path should include the 'bin' directory of your java installation. This is where your system will look for the keytool program.</p> <p>Alternatively you can specify the full path to the keytool to be used e.g. C:\jdk1.6.0_07\bin\keytool</p> <p>Also, please note that if you have multiple versions of java installed, ensure only one is specified in the path.</p> |
| java.net.BindException: Address already in use | |
| Cause: Another service is running on the same port (default 5000) as Xpay4. It could be due to another Xpay4 client already running. | Solution: Stop the other service or run Xpay4 on another port. To use a different port, please see section 2.2.5 Configuration of this document. |
| keytool error: java.lang.Exception: Certificate not imported, alias <merchant_ca> already exists | |
| Cause: The certificate authority 'merchant_ca' has already been added to the keystore, perhaps from an earlier attempt to import the certificates | Solution: Remove the files created during the certificate signing process (site1234.req.pem, site1234.jks, site1234.cert.pem and merchant_ca.cert.pem) and then follow the 2.2.2 Certificate/Key generation section from the start. |
| java.security.cert.CertificateParsingException: signed overrun, bytes = 254 | |
| Cause: There is a problem with one of your Keystores. | Solution: Remove the files created during the certificate signing process (site1234.req.pem, site1234.jks, site1234.cert.pem and merchant_ca.cert.pem) and then follow the 2.2.2 Certificate/Key generation section from the start. |
| javax.net.ssl.SSLHandshakeException: Received fatal alert: handshake failure | |
| Cause: Corrupted certificate. | Solution: The handshake could not be validated and therefore failed. The best course of action on this would be either to create a new directory with a fresh install of XPay4 and generate new certificates OR generate new certificates for the original installation |

6.2 XML Request problems

If you experience problems processing requests via Xpay4 please try following section 4 - An Example Transaction.

Also check the log file created by Xpay4 (default is xpay4.log) for more detailed error information. You may get more information by changing the loglevel setting in the xpay4.ini configuration file (see 2.2.5 Configuration).

The following tables show the cause and solution for common error messages:

| Request aborted: Unable to send to gateway | |
|--|---|
| Cause: This may indicate there is a problem with the certificates. | Solution: Ensure that the certificates you have received from SecureTrading have not been corrupted in any way (for example accidentally editing characters in the file). Try following the Certificate Import section again from the beginning |

| Invalid or incomplete XML java.net.SocketTimeoutException: Read timed out | |
|--|--|
| Cause: This indicates a problem with your XML request. Either it is not well formed or the complete request is not being received by Xpay4 | Solution: Ensure there is a newline at the end of the XML request. Xpay4 will only read complete lines. Ensure all your XML tags are well formed (each tag must be closed and nested tags must be fully enclosed within other tags) Ensure any XML characters are correctly encoded (e.g. '<' must be encoded to '<') |

6.3 Contacting support

If you are unable to solve the problem using the tables above please contact SecureTrading support. Your problem will be solved much more quickly if you have the following information to hand:

- Your Site Reference
- The operating system and java version you are using
- A copy of the Xpay4 logfile with the exact error messages
- A copy of the XML request you are sending and any response XML you receive.

6.4 Xpay4 Characters

Xpay4 requires a valid XML header. All characters must be encoded using the specified encoding. Only characters from the latin-1 (iso-8859-1) character set are currently supported. The most common way to specify this is to ensure that the first line of your XML request is:

```
<?xml version="1.0" encoding="iso-8859-1"?>
```

7 Additional Information

This section contains additional information regarding the Xpay4 client. Please note information regarding the XML to be supplied and the different requests possible is included within the XML Specification for each request type.

7.1 Support

SecureTrading provides support for its software and the operation of its payment service. If you require technical support, first ensure that you have read and understood all relevant documentation.

If the problem persists, please email support@securetrading.com, quoting your SecureTrading alias and concisely stating the nature of your problem.

To help us help you, please include the original XML string sent and any error messages that are returned by the Xpay4 client verbatim.
SecureTrading additional contact details:

Phone: **01248 672 050**

Fax: **01248 672 099**

7.2 Further Reading

More documentation can be found on the SecureTrading website at:

<http://www.securetrading.com>

In particular you may find the following related documents useful:

- ST XML Specification

8 Glossary of terms

8.1 Authorisation

Authorisation is the process of validating a credit card transaction with an acquiring bank. Authorisation allocates the transaction amount on a customer's credit card but no money is debited from the customer's credit card account until after settlement (8.5) is run.

8.2 Good password

A good password is a password containing at least 8 characters from more than two different character types, i.e. lower case letters (d), upper case letters (D), numbers (2), special characters (@).

8.3 Payment gateway

A payment gateway can be thought of as a secure, reliable bridge from a merchant's website or server to the acquiring banks.

8.4 Refund

A refund can only be performed on an authorisation that has settled.

Authorisation is a two stage process. Firstly the authorisation is performed and the funds are allocated in the Customer's account. Secondly, the transaction is settled (8.5) and funds are transferred to the Merchant's account.

In order for a refund to be performed on a transaction, the initial authorisation must have already settled.

8.5 Settlement

Settlement is the process of debiting the transaction amount from a customer's card and into the merchant's account or vice-versa. Typically settlement follows an authorisation or refund request.

8.6 String

A String is a sequence of characters, such as "hello world". The maximum permitted string length when passing an XML string to SecureTrading is 255 characters.

8.7 XML

eXtensible Markup Language provides a structured method of defining data. For further information on XML, please refer to <http://www.xml.org/>