IP Payments

making cash flow

Tokenisation and Customer Registration Integration Guide V 1.2

IP Payments Pty Ltd
Level 3, 441 Kent Street
Sydney
NSW 2000
Australia
(ABN 86 095 635 680)

T +61 2 9255 9500
F +61 2 8248 1276
www.ippayments.com

Table of Contents

1 About this document

1.1 Document history

Version	Date Modified	Author	Summary of Changes
V1.0	01/05/2014	Anne Kehoe	Document created based on API specification.
V1.1	31/10/2014	Anne Kehoe	Content update.
V1.2	27/05/2015	Anne Kehoe	Added details about NZ bank account information.

1.2 Definitions

The following terms and abbreviations are used in this document:

Term	Description
IPP	IP Payments, a premium payments solutions provider uniquely skilled in providing high-quality, efficient and customised solutions to corporate organisations in all industry sectors.
Merchant	For the purposes of this document your company will be referred to as the 'merchant'. A person or company involved in wholesale trade, supplying goods or services to a business or consumer market.
Acquiring Bank	An acquiring bank (or acquirer) is a bank or financial institution that processes credit or debit card payments on behalf of a merchant.
CR	A change request is an issue, defect or new requirement which is raised by a business person and/or representative of a local affiliate which is not described or described in a different way in the latest version of the SDS document (+ amendments).
CC	Creditcard.
Security Code (CVV2/CSC2/CCV)	The Security Code is a 3 or 4 digit code on the back of the cardholder's card. This is used to verify the customer is in possession of the card.
PAN	Primary Account Number (Credit Card Number).
PCI-DSS	Payment Card Industry Data Security Standard. PCI-DSS is an information security standard for organizations that handle cardholder information for the major debit, credit, prepaid, e-purse, ATM, and POS cards.
Access Portal	Access Portal is the platform used by IPP to implement our hosted payment applications – HPP and iHPP.
HPP	The Hosted Payment Page is a standalone payment page which is not integrated into an application.
iHPP	The integrated Hosted Payment Page is integrated into the merchant's website dynamically accepting transaction data prior to the customer entering their card details. Notification of the transaction result is sent back to the merchant in real-time.
iFrame	Inline Frame, a HTML tag used to embed another document within an existing HTML document. Specifically used in this document to describe how to embed the IPP payment page in the merchants website.

DL	Direct Link, this is the value used to specify which iHPP template is to be used. Required where more than one iHPP exists for a particular client account.
HTML	Hypertext Markup Language
POST	A method for sending HTML form data over the Internet. Post data is encoded within the message body.
GET	A method for sending HTML form data over the Internet. Get data is encoded by a browser into the URL.
CSS	Cascading Style Sheets is a style sheet language used for describing the presentation of a web page.
SST	Secure Session Token
URL	Uniform Resource Locator
WSDL	Web Services Description Language
PRM	Payment Relationship Manager, IPP's transaction reporting tool used for user administration, viewing transaction history, refunding and downloading reports among other functionality.
CSV	CSV meaning Comma Separated Values is a report format which can be downloaded from our reporting tool, PRM.
API	Application Programming Interface, a merchant can use IPP's API to gain access to the features and data of our services and applications.
SOAP	*Simple Object Access Protocol, a protocol specification for exchanging structured information in the implementation of web services.*
XML	eXtensible Markup Language, defines a set of rules for encoding documents in a format that is both human-readable and machine-readable.
SIPP	Statement of Invoice Presentment and Payment
TCP/IP	Transmission Control Protocol/Internet Protocol
Tokenisation	Storage of the customer's card data against a unique reference called a token in IPP's secure PCI-DSS compliant system for future use in recurring or one-click payments. This removes the need for the merchant to store card data minimising their PCI DSS scope. This is an additional service and must be enabled on your account.
Token	The unique reference that the customers card data is stored against in IPP's secure PCI-DSS compliant system.
MOD10 Check	A simple algorithm used to validate a credit card number.

2 Purpose of this document

The purpose of this document is to describe the requirements and functionality of the API solution that will be implemented for you by IP Payments Pty Ltd ("IPP"). This document also outlines what is involved for you, the merchant, to integrate into the IPP API. If this document does not meet your needs, please contact us to discuss your requirements and the bespoke solutions we can offer.

3 Introduction

IP Payments is a premium payments solutions provider, uniquely skilled in providing high-quality, efficient, reliable and customised solutions to corporate organisations in all industry sectors. We develop and manage web based billing, payment and reconciliation services for some of the most recognised brand names in the world.

Tokenisation / Customer Registration

IPP offer a card storage solution which allows merchant's to store their customer's card data against a unique reference in IPP's secure PCI-DSS compliant system. The unique reference can subsequently be submitted by the merchant for future recurring or one-click payment requests. Once the token is submitted by the merchant, IPP will look up the corresponding card data and process the transaction as normal. This removes the need for the merchant to store card data in their system minimising their PCI DSS scope and ensuring ongoing compliance against maturing standards.

This document details the options for the initial token creation and the submission methods IPP offer for processing future token payment requests.

API

IPP offer our merchants the ability to securely and efficiently process online, real-time transactions via an API. Some methods discussed in this document refer to the use of IPP's API. The API web service accepts and processes SOAP requests from a remote location over TCP/IP. Transaction results are returned real-time via the API.

4 Process Overview

There are two ways to store a credit card for future use with IPP, tokenisation and customer registration.

- **Tokenisation**
  - With Tokenisation, IPP will generate the token used to store the card. The new token will be used in place of the cardnumber for subsequent payments.
  - The IPP token is generated in a specific format detailed in section IPP Token format.
- **Customer Registration**
  - Customer registration can be used for storage of both the customer's card details and direct debit details.
  - With Customer Registration you will control the token creation by submitting a customer number in the format you wish to use for storage against the card data or direct debit data.
  - The customer number will be used in place of the card number or bank account details for subsequent payments.
  - With this option, you also have the ability to set up a payment schedule for your customer where IPP will manage all subsequent payments. Please contact IPP for further information on this service.

5 Tokenisation

As mentioned above, with Tokenisation IPP will generate the token used to store the card. The new token will be used in place of the cardnumber for subsequent payments.

5.1 IPP Token format

There are a number of token formats which you can use depending on your requirements. By default the below token formats are used. See appendix Token Formats for full list and description of IPP token formats. If you require a different token format to Algorithm 2, please advise IPP prior to your account set up.

5.1.1 Credit Card - Algorithm 2 Token Format

This IPP token is unique for each card and will not display or bear any resemblance to the original credit card number. It will not display any part of the original credit card number in the token (e.g. the last 4 digits). The token is MOD 10 compliant (last digit will be a check digit).

Token format is: **9NNNNNNNNNNNNNNC**

Token format		
Character	**Position**	**Description**
9	1	First digit - The first digit is “9” to differentiate the sequence from any credit card number as no credit card number starts with the digit “9”
NNNNNNNNNNNNNN	2-15	Random numeric string, 14 characters in length.
C	16	Check digit. Used in the MOD10 check.

5.2 Initial Token Creation

There are a number of ways in which you can store your customer’s credit card data for future use. Generally this can be tokenised with the first payment the customer makes to you. You also have the option to tokenise the card without taking payment. There are two integration channels you can use:

1. If you are using IPP’s iHPP payment to accept your customer’s initial transaction / tokenise the card data, see section iHPP Process for further information.
2. If you intend to use IPP’s API to capture the customer’s initial transaction / tokenise the card data, see section API Process for further information.

5.2.1 iHPP Process

IPP’s integrated Hosted Payment Page (iHPP) provides you with the ability to accept credit card payments on your website in real-time. IPP’s iHPP is device agnostic rendering responsively for web, mobile and table devices. You can also use iHPP to store your customer’s card data for future use.

For full information on how to tokenise customers payment data using iHPP, please see the iHPP integration guide which can be provided by IPP at your request.

Please note while initial transactions can be processed via iHPP all subsequent transactions, i.e. once the payment data has been stored, will be processed via the API as described in section Subsequent Token Payments.

5.2.2 API Process

This section provides details on the API requests which can be used to create a token in place of a credit card and process a payment simultaneously or tokenise the card data only. Please note initial tokenisation of direct debit details is not supported via the API integration method.

Please see Appendix

API Overview which provides API information including details on PCI-DSS compliance, system access, SOAP and XML construction and security.

5.2.2.1 New Tokenisation with Payment

For an initial customer payment where you wish to store the customers card details for future use, you can use the SubmitSinglePayment SOAP method with the transaction XML detailed below.

If a credit card has previously been stored, then transaction will be processed and the previously generated CreditCardToken will be returned.

5.2.2.1.1 New Tokenisation with Payment Request

Web Service URL: Secure Remote API
SOAP Method: SubmitSinglePayment

Tokenisation with Payment – XML Element Description					
ELEMENT NAME	**MAX SIZE**	**FORMAT**	**SAMPLE**	**MANDATORY / OPTIONAL**	**DESCRIPTION**
Transaction /Security/UserName	32	AlphaNumeric	Username	M	API Username
Transaction /Security/Password	16	AlphaNumeric	Password	M	API Password
Transaction /AccountNumber	16	Alpha Numeric	123456789	O	This value dictates which account the transaction will be processed through. If this value us not populated, the transaction will be processed to the account tied to the username field.
Transaction /CustomerStorageNumber	32	AlphaNumeric	VAULT1	O	The account within your account structure against which to store the credit card token.
Transaction /MerchantNumber	16	AlphaNumeric	123456789	O	This field is only required if you have multiple bank Merchant ID's set up against one account. This identifier will route the payment to the correct Merchant Number on your account.
Transaction /CustRef	64	AlphaNumeric	Invoice1234	O	An additional reference for the transaction sent by you for reporting purposes.
Transaction /CustNumber	64	AlphaNumeric	Cust1234	O	An additional reference for the transaction sent by you for reporting purposes.
Transaction	20	Numeric	4005550000	M	This field accepts the card number

/CreditCard /CardNumber			000001		which you wish to store. It also accepts a CreditCardToken which has been previously stored.
Transaction /CreditCard /ExpM	2	Numeric	02	M	Expiry Date Month as MM
Transaction /CreditCard /ExpY	4	Numeric	2013	M	Expiry Date Year as YYYY
Transaction /CreditCard /CVN	4	Numeric	123	O	Numeric card verification number
Transaction /CreditCard /CardHolderName	64	Alpha Numeric	John Smith	O	Name as it appears on credit card
Transaction /Amount	10	Numeric	100	M	The amount of the transaction (in cents) e.g. $10.00 = 1000.
Transaction /UserDefined	N/A	N/A	N/A	O	This is an XML node only. User defined elements can be included in the transaction request and used for reporting purposes at a later stage.
Transaction /UserDefined/ Reference1	32	*Text*	YourReference1	O	User defined elements can be included in the transaction request and used for reporting purposes at a later stage.
Transaction /UserDefined/ Reference2	64	Text	YourReference2	O	User defined elements can be included in the transaction request and used for reporting purposes at a later stage.
Transaction /UserDefined/ Reference3	128	Text	YourReference3	O	User defined elements can be included in the transaction request and used for reporting purposes at a later stage.
Transaction /UserDefined/ Reference5	1024	Text	YourReference4	O	User defined elements can be included in the transaction request and used for reporting purposes at a later stage.
Transaction /UserDefined/ Reference10	1024	Text	YourReference5	O	User defined elements can be included in the transaction request and used for reporting purposes at a later stage.
Transaction /CreditCard /TokeniseAlgorithmID	4	Numeric	4	M	This ID defines the token format to use. Please see section IPP Token format for details on the value to use.

5.2.2.1.1.1 Example XML of New Tokenisation with Payment

```
<Transaction>
        <CustomerStorageNumber>VAULT1</CustomerStorageNumber>
        <MerchantNumber >111222333</MerchantNumber>
        <CustRef>123456</CustRef>
        <Amount>5500</Amount>
        <TrnType>1</TrnType>
        <CreditCard >
                <TokeniseAlgorithmID>2</TokeniseAlgorithmID>
                <CardNumber>4005550000000001</CardNumber>
                <ExpM>05</ExpM>
                <ExpY>2013</ExpY>
                <CVN>123</CVN>
                <CardHolderName>John Smith</CardHolderName>
        </CreditCard>
        <Security>
                <UserName>username</UserName>
                <Password>password</Password>
        </Security>
        <UserDefined></UserDefined>
</Transaction>
```

5.2.2.1.2 New Tokenisation with Payment Response
The list below provides an overview of the transaction elements that will be returned in the XML response.

Tokenisation with Payment Response– XML Element Description		
ELEMENT NAME	**MAX SIZE**	**DESCRIPTION**
Response/ResponseCode	1	0 – Approved, 1 = Not Approved
Response/TimeStamp	19 yyyy-MM-dd HH:MM:SS	2010-02-24 21:18:47
Response/Receipt	8	The IPP receipt number generated for this transaction
Response/SettlementDate	8 yyyy-MM-dd	The date which the transaction will be settled with the acquiring bank. This is supplied even when Approved = 0, but can be ignored in this case.
Response/DeclinedCode	3	If result = 0, then this will be blank If result = 1, then this will be the reason for the declined transaction as an error code.
Response/DeclinedMessage	256	If result = 0, then this will be blank If result = 1, then this will be the textual description of the error
Response/CreditCardToken	16	Token generated by IPP to uniquely identify this customer/credit card. (please don't confuse this with the secure session token)
Response/TruncatedCard	16	The truncated credit card. This will show first 6 and last 4 digits of card number, by example: 123456******4321

Response/ExpM	2 MM	The credit card expiry month
Response/ExpY	4 YYYY	The credit card expiry year
Response/CardType	16	e.g. MasterCard, Visa, American Express, Diners

5.2.2.1.2.1 Example Response for New Tokenisation with Payment

```
<Response>
        <ResponseCode>0</ResponseCode>
        <Timestamp>2010-05-23 21:18:47</Timestamp>
        <Receipt>10001197</Receipt>
        <SettlementDate>2010-05-23</SettlementDate>
        <DeclinedCode></DeclinedCode>
        <DeclinedMessage></DeclinedMessage>
        <CreditCardToken>9660066265305097</CreditCardToken>
        <TruncatedCard>123456******1234</TruncatedCard>
        <ExpM>02</ ExpM>
        <ExpY>2010</ExpY>
        <CardType>Visa</CardType>
</Response>
```

5.2.2.2 New Tokenisation Only

5.2.2.2.1 New Tokenisation Only Request

This method is used to submit a credit card number for tokenisation only without payment.

- If the credit card number does not currently exist then card details will be stored, a token will be created and returned in the API response.
- If the credit card number already exists and if an expiry is supplied this will be updated. The original token will be returned in the API response.
- The token can also be submitted in the cardnumber field providing you with the ability to update the expiry date stored against an existing token. See section Updating an Existing Token without Payment.

Web Service URL: SIPP API
SOAP Method: TokeniseCreditCard

Tokenisation Only Request – XML Element Description

ELEMENT NAME	MAX SIZE	FORMAT	SAMPLE	MANDATORY / OPTIONAL	DESCRIPTION
UserName	32	AlphaNumeric	Username	M	API Username
Password	16	AlphaNumeric	Password	M	API Password
CustomerStorageNumber	32	AlphaNumeric	VAULT1	O	The account within your account structure against which to store the credit card token.
CardNumber	20	Numeric	4005550000000001	M	This field accepts the card number which you wish to store.

					It also accepts a CreditCardToken which has been previously stored allowing you to update the expiry date stored against the Token.
ExpM	2	Numeric	02	M	Expiry Date Month as MM
ExpY	4	Numeric	2013	M	Expiry Date Year as YYYY
TokeniseAlgorithmID	4				This ID that defines the token format to use. By default this needs to be set to "2".

5.2.2.2.1.1 Example XML of TokeniseCreditCard request

```
<TokeniseCreditCard>
        <UserName>username</UserName>
        <Password>password</Password>
        <CustomerStorageNumber>VAULT1</CustomerStorageNumber>
        <CardNumber>4242424242424242</CardNumber>
        <ExpM>02</ExpM>
        <ExpY>2013</ExpY>
        <TokeniseAlgorithmID>2</TokeniseAlgorithmID>
</TokeniseCreditCard>
```

5.2.2.2.2 New Tokenisation Only Response

Please see list of XML elements below for the response.

Tokenisation Only Response – XML Element Description

ELEMENT NAME	MAX SIZE	DESCRIPTION
ReturnValue	1	0 - Successful 1 - Invalid username/password 4 - Invalid CustomerStorageNumber 5 - Invalid Credit Card Number 99 - Exception encountered
Token	16	9123456789012345

5.2.2.2.2.1 Example XML of TokeniseCreditCard response

```
<TokeniseCreditCardResponse>
        <ReturnValue>0</ReturnValue>
        <Token>9123456789123456</Token>
</TokeniseCreditCardResponse>
```

5.3 Subsequent Token Payments

Once you have carried out an initial transaction for a customer and stored the card details against a token reference, you can process future payment requests using the stored token in the following scenarios:

Stored payment method

1. You can enhance your returning customers experience by presenting them with a masked version of their card number stored through a previous payment. If the customer decides to use the same payment method, you can submit the payment request using the API request in section Single Payment on Existing Token.
2. If the customer decides they want to use a new card, you can send them to IPP's iHPP / your API integration to enter their new card details.

Recurring payments

3. You may require the ability to set up your customers for recurring payments for example if the customer needs to pay a monthly subscription.
4. IPP provide a number of integration methods for recurring payments with tokenisation such as
    a. Submission via API – See section Single Payment on Existing Token (Only available for stored credit cards).
    b. Submission via batch – See section Bulk Payments on Existing Token.

Note: IPP also provide you with the ability to create a schedule of payments for a customers stored payment details. To do this you must use customer registration rather than tokenisation. Please see section Customer Registration.

5.3.1 Single Payment on Existing Token

To process a payment using a token that has already been created you can use the following request.

- This transaction request can be used to process a single payment where the customer has returned to your site.
- It can also be used if you need to process multiple recurring payments; you can stream these XML requests through to IPP and receive a response for each transaction in real time.

Web Service URL: Secure Remote API
SOAP Method: SubmitSinglePayment

5.3.1.1 Example XML of Single Payment on Existing Token Request

```
<Transaction>
        <CustomerStorageNumber>VAULT1</CustomerStorageNumber>
        <MerchantNumber >111222333</MerchantNumber >
        <CustRef>123456</CustRef>
        <Amount>5500</Amount>
        <TrnType>1</TrnType>
        <CreditCard >
                <TokeniseAlgorithmID>2</TokeniseAlgorithmID>
                <CardNumber>9660066265305097</CardNumber>
        </CreditCard>
        <Security>
                <UserName>username</UserName>
                <Password>password</Password>
        </Security>
        <UserDefined></UserDefined>
        </Transaction>
```

5.3.1.2 Example XML of Single Payment on Existing Token Response

```
        <Response>
                <ResponseCode>0</ResponseCode>
                <Timestamp>2010-05-23 21:18:47</Timestamp>
                <Receipt>10001197</Receipt>
                <SettlementDate>2010-05-23</SettlementDate>
                <DeclinedCode></DeclinedCode>
                <DeclinedMessage></DeclinedMessage>
                <CreditCardToken>9660066265305097</CreditCardToken>
                <TruncatedCard>123456******1234</TruncatedCard>
                <ExpM>02</ExpM>
                <ExpY>2010</ExpY>
                <CardType>Visa</CardType>
        </Response>
```

5.3.2 Bulk Payments on Existing Token

IPP provide an option to upload multiple payments in a batch file.

- Using a pre-defined file format, you can create a batch file of payments containing the previously created tokens and the payment amount you wish to take from each token.
- The file can be uploaded automatically via the API or manually through IPP's transaction management tool, PRM.
- Once the file has been processed, IPP will return a response file in a pre-defined format which details the response for each transaction.

Please contact IPP for further information on this service.

5.4 Updating an Existing Token

5.4.1 Updating an Existing Token with Payment

Use this request to update the details of an existing token while also taking a payment.

Web Service URL: Secure Remote API
SOAP Method: SubmitSinglePayment

5.4.1.1.1 Example XML of Token Update Request

```
        <Transaction>
                <CustomerStorageNumber>VAULT1</CustomerStorageNumber>
                <MerchantNumber >111222333</MerchantNumber >
                <CustRef>123456</CustRef>
                <Amount>5500</Amount>
                <TrnType>1</TrnType>
                <CreditCard >
```

```
                        <TokeniseAlgorithmID>2</TokeniseAlgorithmID>
                        <CardNumber>9660066265305097</CardNumber>
                        <ExpM>05</ExpM>
                        <ExpY>2013</ExpY>
                </CreditCard>
                <Security>
                        <UserName>username</UserName>
                        <Password>password</Password>
                </Security>
                <UserDefined></UserDefined>
        </Transaction>
```

5.4.1.2 Example XML of Token Update Response

Please see example XML response for a Token update below.

```
        <Response>
                <ResponseCode>0</ResponseCode>
                <Timestamp>2010-05-23 21:18:47</Timestamp>
                <Receipt>10001197</Receipt>
                <SettlementDate>2010-05-23</SettlementDate>
                <DeclinedCode></DeclinedCode>
                <DeclinedMessage></DeclinedMessage>
                <CreditCardToken>9660066265305097</CreditCardToken>
                <TruncatedCard>123456******1234</TruncatedCard>
                <ExpM>02</ExpM>
                <ExpY>2010</ExpY>
                <CardType>Visa</CardType>
        </Response>
```

5.4.2 Updating an Existing Token without Payment

This is the same request as specified in section New Tokenisation Only Request.
Use this request to update the details of an existing token without taking a payment. The token can be submitted in the cardnumber field of the request providing you with the ability to update the expiry date stored against an existing token.

Web Service URL: SIPP API
SOAP Method: TokeniseCreditCard

5.4.2.1 Example XML of Token Update Request

```
        <TokeniseCreditCard>
                <UserName>username</UserName>
                <Password>password</Password>
                <CustomerStorageNumber>VAULT1</CustomerStorageNumber>
                <CardNumber>9660066265305097</CardNumber>
                <ExpM>02</ExpM>
```

```
            <ExpY>2013</ExpY>
            <TokeniseAlgorithmID>2</TokeniseAlgorithmID>
    </TokeniseCreditCard>
```

5.5 Removing an Existing Token

5.5.1 Removing an Existing Token Request

This method can be used to delete (disassociate) the token that has been issued by IPP systems. When this function is called IPP will only delete the credit card details, the token remains so that if this credit card is added again in the future, the same token will be returned in the response. The token is also maintained so that a different credit card will not be issued with the same token.

Web Service URL: SIPP API
SOAP Method: DeleteCreditCardForToken

Remove and Existing Token Request		
ELEMENT NAME	**MAX SIZE**	**DESCRIPTION**
UserName	32	API Username
Password	16	API Password
Token	16	The original token issued by the IPP systems. i.e. 9123456789012345

5.5.1.1 Example XML for Existing Token Request

```
    <DeleteCreditCardForToken>
            <UserName>username</UserName>
            <Password>password</Password>
            <CCToken>9123456789123456</CCToken>
    </DeleteCreditCardForToken>
```

5.5.2 Removing an Existing Token Response

Remove and Existing Token Response		
ELEMENT NAME	**MAX SIZE**	**DESCRIPTION**
ReturnValue	1	0 - Successful 1 - Invalid username/password 2 - Invalid token 3 - No credit card details stored for supplied token 99 - Exception encountered
Token	16	The token that has been deleted/disassociated. i.e. 9123456789012345

5.5.2.1 Example XML for Existing Token Response

The response is a simple XML document which looks as follows.

```
<DeleteCreditCardResponse>
        <ReturnValue>1</ReturnValue>
        <Token>9123456789123456</Token>
</DeleteCreditCardResponse>
```

6 Customer Registration

Customer registration allows customer details including credit card data, to be registered within IP Payments at the time a transaction is being processed. This ensures secure storage of payment information within IP Payments' secure PCI compliant environment.

This is very similar to our tokenisation solution; however in this method of card storage customer data is stored against the unique reference CustNumber. This allows you to generate the token and use any format you wish as long as it is unique. There is a one to one relationship between CustNumber and card data i.e. only one card number can be stored against one CustNumber.

6.1 Initial Customer Registration

There are a number of ways in which you can store your customer's credit card data for future use. Generally this can be registered with the first payment the customer makes to you. You also have the option to register the card without taking payment. There are two integration channels you can use:

1. If you are using IPP's iHPP to accept your customer's initial transaction / register the card data, see section iHPP Process for further information.
2. If you intend to use IPP's API to capture the customer's initial transaction / register the card data, see section API Process for further information.

6.1.1 iHPP Process

IPP's integrated Hosted Payment Page (iHPP) provides you with the ability to accept credit card payments on your website in real-time. The iHPP is device agnostic rendering responsively for web, mobile and table devices. You can also use iHPP to store your customer's card data for future use.

For full information on how to register customers card data using the iHPP, please see the iHPP integration guide which can be provided by IPP at your request.

Please note while initial transactions can be processed via iHPP, all subsequent transactions i.e. once the card data has been stored, will be processed via the API as described in section Subsequent Customer Registration Payments.

Processing subsequent tokenised transactions via the API does not add to the scope of your PCI-DSS requirements as sensitive card data will be stored with the initial payment via iHPP.

6.1.2 API Process

This section provides details on the API requests which can be used to register a customer and store their card data against a customer number.

Please see Appendix

API Overview which provides API information including details on PCI-DSS compliance, system access, SOAP and XML construction and security.

Once a customer has been registered within IP Payments, you can then use the options described in Section New Customer Registration Only to take further payments. It is essential that data supplied is correct and conforms to requirements, otherwise customer data will not be saved and the transaction will be rejected with Declined Code 122 (Registered customer details not found).

6.1.2.1 New Customer Registration with Payment

6.1.2.1.1 New Customer Registration with Payment Request

The following list provides an overview of available customer detail elements, all of which are logged to the IP Payments database.

Please note that the CustNumber must appear twice in the customer payment/registration XML – once associated with the payment, and once associated with the customer.

The additional transaction fields which should be included in the SubmitSinglePayment method are listed below. Please note you must also include the normal payment transaction fields.

Web Service URL: Secure Remote API
SOAP Method: SubmitSinglePayment

SubmitSinglePayment with customer registration – XML Element Description

ELEMENT NAME	MAX SIZE	FORMAT	SAMPLE	MANDATORY / OPTIONAL	DESCRIPTION
Register/Customer/AccountNumber	16	Alpha Numeric	123456789	O	This value dictates which account the transaction will be processed through. If this value is not populated, the transaction will be processed to the account tied to the username field.
Register/Customer/CustNumber	32	Alpha Numeric	12345678	M	Merchant assigned unique customer number used to store the customer data and card data against.
Register/Customer/Contact/FirstName	32	Alpha Numeric	John	M	Customer first name
Register/Customer/Contact/LastName	32	Alpha Numeric	Smith	M	Customer second name
Register/Customer/CreditCard	-	-	-	M	XML tag containing card data fields. This field has one attribute, 'Registered'. This attribute is used to define: 1. If you are registering a customer's data with IPP for future use. Registered attribute should be set to 'True' if registering a credit card against a new customer. The card data fields are mandatory in this scenario. 2. Processing a payment on a previously registered customer. Registered attribute should be set to 'True' if using a previously

					registered customer's credit card. In this scenario the card data fields are not required. 3. If processing a payment for an unregistered customer / payment method. Registered attribute should be set to 'False' in this scenario and card data fields must be sent.
Register/Customer/CreditCard/CardNumber	20	Numeric	4005550000000001	O	Used to send the credit card number where required. See scenarios under Register/Customer/CreditCard field.
Register/Customer/CreditCard/ExpM	2	Numeric	05	O	Used to send the credit card expiry month where required. See scenarios under Register/Customer/CreditCard field.
Register/Customer/CreditCard/ExpY	4	Numeric	2013	O	Used to send the credit card expiry year where required. See scenarios under Register/Customer/CreditCard field.
Register/Customer/CreditCard/CardHolderName	64	Alpha Numeric	John Smith	O	Used to send the credit card owner name where required. See scenarios under Register/Customer/CreditCard field.
Register/Customer/BankAccount/					XML tag containing bank account fields. This field has one attribute, 'Registered'. This attribute is used to define: 1. If you are registering a customer's data with IPP for future use. Registered attribute should be set to 'True' if registering a bank account against a new customer. The bank account fields are mandatory in this scenario. 2. Processing a payment on a previously registered customer. Registered attribute should be set to 'True' if using a previously registered customer's bank account. In this scenario the bank account fields are not required. 3. If processing a payment for an unregistered customer / payment method. Registered attribute should be set to 'False' in this

					scenario and bank account fields must be sent.
Register/Customer/BankAccount/AccNo	24	Numeric	123456789	O	Bank account number, submitted where required. See scenarios under Register/Customer/BankAccount field. For NZ bank details: The NZ Bank Account and Suffix should be included in this field with no spaces.
Register/Customer/BankAccount/AccRouting	16	Alpha Numeric	012-123	O	Bank account routing, submitted where required. See scenarios under Register/Customer/BankAccount field. For NZ bank details: The NZ Bank and Branch codes should be included in this field with no spaces.
Register/Customer/BankAccount/AccTitle	64	Alpha Numeric	John Smith Pty Ltd	O	Bank account title – the name on the bank account, submitted where required. See scenarios under Register/Customer/BankAccount field.
Register/Customer/BankAccount/BankName	32	Alpha Numeric	Bank XYZ	O	Name of the bank/financial institution where the account is held, submitted where required. See scenarios under Register/Customer/BankAccount field.
Register/Customer/BankAccount/BankSuburb	32	Alpha Numeric	Sydney	O	Suburb where the bank/financial institution where the account is held, submitted where required. See scenarios under Register/Customer/BankAccount field.
Register/Customer/BankAccount/BankState	32	Alpha Numeric	NSW	O	State where the bank/financial institution where the account is held, submitted where required. See scenarios under Register/Customer/BankAccount field.
Register/CustUserDefined	-	-	-	-	Please see IP Payments regarding the storage of additional data against a customer
Register /CustUserDefined/Reference1	32	Text	YourReference1	O	User defined elements can be included in the customer registration request and used for reporting purposes at a later stage.

Register/CustUserDefined/Reference2	64	Text	YourReference2	O	User defined elements can be included in the customer registration request and used for reporting purposes at a later stage.
Register/CustUserDefined/Reference3	128	Text	YourReference3	O	User defined elements can be included in the customer registration request and used for reporting purposes at a later stage.
Register/CustUserDefined/ Reference4	1024	Text	YourReference4	O	User defined elements can be included in the customer registration request and used for reporting purposes at a later stage.
Register/CustUserDefined/Reference5	1024	Text	YourReference5	O	User defined elements can be included in the customer registration request and used for reporting purposes at a later stage.

6.1.2.1.1.1 Example XML for new customer registration with a credit card payment

```
<Transaction>
        <CustNumber>12345678</CustNumber>
        <CustRef>123456</CustRef>
        <Amount>5500</Amount>
        <TrnType>1</TrnType>
        <CreditCard Registered="True">
        </CreditCard>
        <Security>
                <UserName>username</UserName>
                <Password>password</Password>
        </Security>
        <UserDefined>
                <Reference1>123</Reference1>
       </UserDefined>
        <Register>
                <Customer>
                        <AccountNumber>Account1</AccountNumber>
                        <CustNumber>12345678</CustNumber>
                        <Contact>
                                <FirstName>Bill</FirstName>
                                <LastName>Smith</LastName>
                        </Contact>
                        <CreditCard>
                                <CardNumber>4005550000000001</CardNumber>
                                <ExpM>05</ExpM>
                                <ExpY>2013</ExpY>
                                <CardHolderName>John Smith</CardHolderName>
                        </CreditCard>
                        <CustUserDefined>
```

```
                                        <Reference1>YourReference</Reference1>
                                </CustUserDefined>
                        </Customer>
                </Register>
        </Transaction>
```

6.1.2.1.1.2 Example XML for new customer registration with a direct debit payment

```
        <Transaction>
                <CustNumber>12345678</CustNumber>
                <CustRef>123456</CustRef>
                <Amount>5500</Amount>
                <TrnType>1</TrnType>
                <CreditCard Registered="True">
                </CreditCard>
                <Security>
                        <UserName>username</UserName>
                        <Password>password</Password>
                </Security>
                <UserDefined>
                        <Reference1>123</Reference1>
               </UserDefined>
                <Register>
                        <Customer>
                                <AccountNumber>Account1</AccountNumber>
                                <CustNumber>12345678</CustNumber>
                                <Contact>
                                        <FirstName>Bill</FirstName>
                                        <LastName>Smith</LastName>
                                </Contact>
                        < BankAccount>
                                 <AccNo>098124564</AccNo>
                                 <AccRouting>013259</AccRouting>
                                 <AccTitle>John Smith</AccTitle>
                                 <BankName>ABCBank</ BankName>
                                 <BankSuburb>Smithtown</ BankSuburb>
                                 <BankState>NSW</ BankState>
                         </ BankAccount>
                      <CustUserDefined>
                                <Reference1>YourReference</Reference1>
                        </CustUserDefined >
                        </Customer>
                </Register>
        </Transaction>
```

6.1.2.1.2 New Customer Registration with Payment Response

The list below provides an overview of the transaction elements that will be returned in the XML response.

Remove and Existing Token Response

ELEMENT NAME	MAX SIZE	DESCRIPTION
ReturnValue	1	ReturnValue: 0 - Register Successful 1 - Invalid username/password 2 - Invalid Data Received (in this case, ReturnMessage will contain a textual description of the invalid data) 99 - Exception encountered
ReturnMessage	32	This will be blank for all return values except for 2. In this case, a textual description of the invalid data will appear in this element.
ActionCode	1	1 – New Registration 2 – Update of existing customer registration

6.1.2.1.2.1 Example XML for New Customer Registration with Payment

```
<RegisterSingleCustomerResponse>
            <ReturnValue>0</ReturnValue>
            <ReturnMessage></ReturnMessage>
             <CustomerID>998774231</CustomerID>
             <CustNumber>12345678</CustNumber>
            <ActionCode>1</ActionCode>
</ RegisterSingleCustomerResponse>

<Response>
            <ResponseCode>0</ResponseCode>
            <Timestamp>2010-05-23 21:18:47</Timestamp>
            <Receipt>10001197</Receipt>
            <SettlementDate>2010-05-23</SettlementDate>
            <DeclinedCode></DeclinedCode>
            <DeclinedMessage></DeclinedMessage>
            <CreditCardToken>9660066265305097</CreditCardToken>
            <TruncatedCard>123456******1234</TruncatedCard>
            <ExpM>02</ExpM>
            <ExpY>2010</ExpY>
            <CardType>Visa</CardType>
</Response>
```

6.1.2.2 New Customer Registration Only

The same SOAP method is used for both registering a new customer without submitting a payment and updating an existing customer's details. Please see section Updating a Registered Customer Number for further information.

6.2 Subsequent Customer Registration Payments

Once you have carried out an initial transaction for a customer and stored the card details against ther Customer Number, you can process future payment requests using the stored Customer Number in the following scenarios:

Stored payment method

1. You can enhance your returning customers experience by presenting them with a masked version of their card number / account number stored through a previous payment. If the customer decides to use the same payment method, you can submit the payment request using the API request in section Single Payment On Registered Custumer Number.
2. If the customer decides they want to use a new card, you can send them to IPP's iHPP / your API integration to enter their new card details.

Recurring payments

3. You may require the ability to set up your customers for recurring payments for example if the customer needs to pay a monthly subscription.
4. IPP provide a number of integration methods for recurring payments with Customer Registration such as:
   a. Submission via API – See section Single Payment On Registered Custumer Number.
   b. Submission via batch – See section Bulk Payments on Registered Customer Number.
   c. Creation of a payments schedule with IPP's scheduler.

6.2.1 Single Payment On Registered Custumer Number

To process a payment using a customer number that has already been registered with a payment method you can use the following request.

- This transaction request can be used to process a single payment where the customer has returned to your site.
- It can also be used if you need to process multiple recurring payments; you can stream these XML requests through to IPP and receive a response for each transaction in real time.

Web Service URL: Secure Remote API
SOAP Method: SubmitSinglePayment

6.2.1.1 Example XML for Registered Credit Card Payment

```
<Transaction>
        <CustNumber>12345678</CustNumber>
        <CustRef>123456</CustRef>
        <Amount>5500</Amount>
        <TrnType>1</TrnType>
        <CreditCard Registered=”True”>
        </CreditCard>
        <Security>
                <UserName>username</UserName>
                <Password>password</Password>
        </Security>
        <UserDefined></UserDefined>
        <TrnSource>192.168.0.1</TrnSource>
</Transaction>
```

6.2.1.2 Example XML for Registered Direct Debit Payment

```
<Transaction>
        <CustNumber>12345678</CustNumber>
        <CustRef>123456</CustRef>
        <Amount>5500</Amount>
        <TrnType>7</TrnType>
        <DirectEntry Registered="True">
        </DirectEntry>
        <Security>
                <UserName>username</UserName>
                <Password>password</Password>
        </Security>
        <UserDefined></UserDefined>
        <TrnSource>192.168.0.1</TrnSource>
</Transaction>
```

6.2.2 Bulk Payments on Registered Customer Number

IPP provide an option to upload multiple payments in a batch file.

- Using a pre-defined file format, you can create a batch file of payments containing previously registered customer numbers and the payment amount you wish to take from each customer number.
- The file can be uploaded automatically via the API or manually through IPP's transaction management tool, PRM.
- Once the file has been processed, IPP will return a response file in a pre-defined format which details the response for each transaction.

Please contact IPP for further information on this service.

6.3 Updating a Registered Customer Number

This method is to be used to register a new customer without payment and also to update a customer credit card / bank account details where an existing Customer Number is provided. If the CustNumber provided in this request does not exist in IPP Customer database, a new Customer will be added.

Only elements and associated values included in the method call will be updated.

Web Service URL: SIPP API
SOAP Method: RegisterSingleCustomer

6.3.1 Example XML for Updating a Registered Customer Number

Update the expiry date on an existing card.

```
<RegisterSingleCustomer>
        <Register>
                <Customer>
                        <CustNumber>12345678</CustNumber>
                        <CreditCard>
                                <ExpM>06</ExpM>
                        <ExpY>2015</ExpY>
                    </CreditCard>
                </Customer>
                <Security>
                    <UserName>username</UserName>
                    <Password>password</Password>
                </Security>
            </Register>
        </RegisterSingleCustomer>
```

Add new card details to an existing customer.

```
        <RegisterSingleCustomer>
            <Register>
                <Customer>
                     <CustNumber>12345678</CustNumber>
                    <CreditCard>
                        <CardNumber>4444333322221111</CardNumber>
                        <ExpM>06</ExpM>
                        <ExpY>2015</ExpY>
                        <CardHolderName>John Smith</CardHolderName>
                    </CreditCard>
                </Customer>
                <Security>
                     <UserName>username</UserName>
                    <Password>password</Password>
                </Security>
            </Register>
        </RegisterSingleCustomer>
```

6.3.2 Example Response XML for Updating a Registered Customer Number

The list below provides an overview of the transaction elements that will be returned in the XML response.

Remove and Existing Token Response

ELEMENT NAME	MAX SIZE	DESCRIPTION
ReturnValue	1	ReturnValue: 0 - Register Successful 1 - Invalid username/password 2 - Invalid Data Received (in this case, ReturnMessage will contain a textual description of the invalid data) 99 - Exception encountered
ReturnMessage	32	This will be blank for all return values except for 2. In this case, a textual description of the invalid data will appear in this element.

ActionCode	1	1 – New Registration 2 – Update of existing customer registration

XML response example:

```
<RegisterSingleCustomerResponse>
            <ReturnValue>0</ReturnValue>
            <ReturnMessage></ReturnMessage>
             <CustomerID>998774231</CustomerID>
             <CustNumber>12345678</CustNumber>
            <ActionCode>1</ActionCode>
</ RegisterSingleCustomerResponse>
```

7 Card Data Migration

If you are currently storing card data in your system or a third parties system and would like to move to IP Payments for transaction processing and tokenisation, you will need to carry out a card data migration. The options for card data migration are as follows:

1. IPP provide the ability to upload your existing card data for storage in a one-off bulk file through the API or our transaction management and reporting tool, PRM.
2. You can carry out a once off exercise of streaming your existing customer's payment data to IPP via the API as described in section New Tokenisation Only.

Please contact IPP for further information on migrating your customer's payment data and which method is best for your.

8 Appendix

8.1 Reference Documents

See other guides below which may be useful in implementation of your solution. You can request these guides from IPP.

Document Name	Description
PRM User Guide	This document provides a guide to the functionality available in IP Payments reporting tool, Payment Relationship Manager (PRM).
iHPP Integration Guide	This document outlines the technical implementation required for integrating into IPP’s integrated Hosted Payment Page (iHPP).
API Integration Guide	This document describes the technical details required for integration using IPP’s API.
Batch Payments Guide	This document provides an overview and technical information required for processing batch payments i.e. the submission of multiple payments in a specified file format.
Data Migration Guide	This document details the method used for migration of data such as card data to IPP.

8.2 Token Formats

8.2.1 Credit Card Token Formats

8.2.1.1 Algorithm 1 Token Format

This Token format is comprised of the first 4 & last 4 digits of CC, with middle 8 digits being randomly generated. This token will fail a Luhn check.

TokeniseAlgorithmID: 1
Token Format is **AAAAXXXXXXXXCCCC**

Token format	
Character / String	**Description**
AAAA	First 4 characters will be the first 4 digits of the credit card
XXXXXXXX	The middle 8 characters will be a randomly generated digits
CCCC	Last 4 characters will be the last 4 digits of the credit card

8.2.1.2 Algorithm 2 Token Format

This IPP token is unique for each card and will not display or bear any resemblance to the original credit card number. It will not display any part of the original credit card number in the token (e.g. the last 4 digits). The token is MOD 10 compliant (last digit will be a check digit).

TokeniseAlgorithmID: 2
Token format is: **9NNNNNNNNNNNNNNC**

Token format	
Character / String	**Description**
9	First digit
NNNN	Random numeric string with 14 characters in length
C	Check digit.

8.2.1.3 Algorithm 3 Token Format

This token format provides an indicator of the card scheme stored.

TokeniseAlgorithmID: 3
Token format is: **NNXA AAAA AAAA MMMM**

Token format	
Character / String	**Description**
NN	First 2 digits of the card number being stored.
X	Either V,M,A,D to represent card type • V = Visa • M = MasterCard • D = Diners • A = Amex
AAAAA A AAAA	A Random alpha-numeric string (9 characters in length)

MMMM	Last 4 digits of a given credit card number

8.2.1.4 Algorithm 4 Token Format

TokeniseAlgorithmID: 4
Token Format is:

- Visa: **991nnnnnnnnnnnn** (16 digits)
- MasterCard: **992nnnnnnnnnnnn** (16 digits)
- Amex: **993nnnnnnnnnnn** (15 digits)
- Diners: **994nnnnnnnnnn** (14 digits)

Token format	
Character / String	**Description**
99	First two digits of the card number being stored.
1,2,3 or 4	Third digit identifies card type:- • 1 = Visa • 2 = MasterCard • 3 = Amex • 4 = Diners
n	Random digits

8.2.1.5 Algorithm 5 Token Format

This token format is 16 characters in length for ALL card types (Visa, MasterCard, Amex and Diners).

TokeniseAlgorithmID: 5
Token Format is: **NNNNNNXXXXXXMMMM**

Token format	
Character / String	**Description**
NNNNNN	First 6 characters will be the first 6 digits of the credit card
XXXXXX	The middle 6 characters will be a random alpha-numeric string. At least one Alpha and at least one Number in the random string. Only upper case letters will be used.
MMMM	Last 4 characters will be the last 4 digits of the credit card

8.2.1.6 Algorithm 6 Token Format

This token format is similar to algorithm 1 however the number of characters of the token matches the number of digits of the Card Program. This token format does not pass the Luhn check.

TokeniseAlgorithmID: 6
Token Format is:

- Visa - **NNNNxxxxxxxxMMMM**
- Amex - **NNNNxxxxxxxMMMM**
- Diners – **NNNNxxxxxxMMMM**

Token format

Character / String	Description
NNNN	First 4 characters will be the first 4 digits of the credit card.
XXXXXXXX	For: • Visa and MasterCard, the middle 8 characters will be randomly generated digits. • Amex, the middle 7 characters will be randomly generated. • Diners, the middle 6 characters will be randomly generated.
MMMM	Last 4 characters will be the last 4 digits of the credit card.

8.2.1.7 Algorithm 7 Token Format

This token format is identical to Algorithm 2 however this format will tokenise the card based on a combination of credit card number and the customer number (custnumber field). For example:

- *Same credit card number, different customer numbers:*
  - Customer Number 1 + CC 1
  - Customer Number 2 + CC 1
  - *Result*: Two different tokens are generated
- *Same Customer Number, different credit card numbers:*
  - Customer Number 1 + CC 1
  - Customer Number 1 + CC 2
  - *Result*: Two different tokens are generated

TokeniseAlgorithmID: 7
Token format is: **9NNNNNNNNNNNNNNC**

Token format	
Character / String	**Description**
9	First digit
NNNN	Random numeric string with 14 characters in length
C	Check digit.

8.3 API Overview

8.3.1 PCI-DSS Compliance

IPP adhere to the highest standard of PCI-DSS compliance - Level 1. It can be a common misconception that if you are processing online payments with a secure payment gateway you are automatically PCI-DSS compliant. This is not the case, PCI-DSS applies to all organisations which store, process or transmit cardholder information. How you manage your payment process will define the level of PCI-DSS that you must adhere to.

Please note, using IPP's API to capture and transmit credit card data will bring your website, development process and internal systems in to scope for PCI-DSS.

IPP offer solutions that will allow you to capture and transmit credit card data without adding additional PCI-DSS scope to your business. Please contact IPP for further information on these products and services.

8.3.2 System access

IP Payments has two locations for API web services, Secure Remote API and SIPP API, each has their own set of web service URL's. Each section above documents which web service to call when submitting the transaction to IPP.

8.3.2.1 Secure Remote API URL

The Secure Remote API web service employs the following business logic for processing online transactions:

- A remote application connects and authenticates with IPP's server.
- The transaction data is passed to the API in a SOAP request with a number of parameters required for processing.
- IPP carry out the action required for the API request called and transactional data is logged to the IPP database.
- The API responds with result data to the remote application.

The above process occurs for each API call.

IP Payments supports a test and live web service, each of which can be found at the following locations:

- The test web service URL is located at: https://demo.ippayments.com.au/interface/api/dts.asmx
- The live web service URL is located at: https://www.ippayments.com.au/interface/api/dts.asmx

At these locations, you will also find sample SOAP requests and responses, as well as the WSDL.

8.3.2.2 SIPP API URL

IP Payments supports a test and live web service, each of which can be found at the following locations:

- The test web service URI is located at: https://demo.ippayments.com.au/interface/api/sipp.asmx
- The live web service URI is located at: https://www.ippayments.com.au/interface/api/sipp.asmx

8.3.3 SOAP and XML Construction

IPP use the SOAP protocol to exchange structured XML messages for transaction processing. Payment request and response XML transactions are submitted as a parameter in the SOAP message.

The following is a sample SOAP 1.1 request and response. The **PLACEHOLDERS** shown need to be replaced with actual values. The <trnXML> field will hold the transaction XML that will advise IPP of what action to take.

```
POST /interface/api/dts.asmx HTTP/1.1
Host: www.ippayments.com.au
Content-Type: text/xml; charset=utf-8
Content-Length: LENGTH
SOAPAction: IPP WEB SERVICES URL

<?xml version="1.0" encoding="utf-8"?>
<soap:Envelope xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/">
  <soap:Body>
    <SOAP METHOD xmlns=IPP WEB SERVICES URL>
      <trnXML>STRING</trnXML>
    </SubmitSinglePayment>
  </soap:Body>
</soap:Envelope>
HTTP/1.1 200 OK
Content-Type: text/xml; charset=utf-8
Content-Length: LENGTH



<?xml version="1.0" encoding="utf-8"?>
<soap:Envelope xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/">
  <soap:Body>
    < SOAP METHODResponse xmlns=IPP WEB SERVICES URL>
      < SOAP METHODResult>STRING</SubmitSinglePaymentResult>
    </ SOAP METHODResponse>
  </soap:Body>
</soap:Envelope>
```

The transaction messages discussed in the rest of this document will define the XML which must be submitted in the SOAP message. Please note that XML tags within each API parameter are case sensitive.

Please see an example of the transaction XML below. You must ensure you are passing the transaction XML as a parameter in the SOAP message. If your application is not handling this correctly, you can achieve this through the use of the CDATA tag as shown in the below example, the additional CDATA tags are highlighted in bold red.

Please note that this is valid for all API functions throughout this document.

```
<![CDATA[

   <Transaction>
        <CustNumber>12345678</CustNumber>
        <CustRef>123456</CustRef>
        <Amount>5500</Amount>
        <TrnType>1</TrnType>
        <CreditCard Registered=”False”>
                <CardNumber>4005550000000001</CardNumber>
                <ExpM>05</ExpM>
                <ExpY>2013</ExpY>
                <CVN>123</CVN>
                <CardHolderName>John Smith</CardHolderName>
        </CreditCard>
        <Security>
                <UserName>username</UserName>
                <Password>password</Password>
        </Security>
        <TrnSource>192.168.0.1</TrnSource>
   </Transaction>

]]>
```

8.3.4 Security

Transactions are processed via an industry standard secure https connection. This means the merchant can be assured when the transaction is in transit:

1. Sensitive data such as credit card numbers will remain confidential because all information is 128bit encrypted; and
2. The server the merchant is connecting to will be authenticated as belonging to IP Payments through PKI certificates issued by a root Certificate Authority.

In addition to the above security, each merchant transaction request received by IP Payments is authenticated via a pre-allocated User Id and Password, both 128bit encrypted.

8.4 Bank Response Codes

Code	Response Text	Code	Response Text
APPROVED			
00	Approved	08	Honour with ID
11	Approved VIP (not used)	16	Approved, Update Track 3 (not used)
77	Approved (ANZ only)		
DECLINED			
01	Refer to Card Issuer	41	Lost Card—Pick Up
02	Refer to Issuer's Special Conditions	42	No Universal Amount
03	Invalid Merchant	43	Stolen Card—Pick Up
04	Pick Up Card	44	No Investment Account
05	Do Not Honour	51	Insufficient Funds
06	Error	52	No Cheque Account
07	Pick Up Card, Special Conditions	53	No Savings Account
09	Request in Progress	54	Expired Card
10	Partial Amount Approved	55	Incorrect PIN
12	Invalid Transaction	56	No Card Record
13	Invalid Amount	57	Trans. not Permitted to Cardholder
14	Invalid Card Number	58	Transaction not Permitted to Terminal
15	No Such Issuer	59	Suspected Fraud
17	Customer Cancellation	60	Card Acceptor Contact Acquirer
18	Customer Dispute	61	Exceeds Withdrawal Amount Limits
19	Re-enter Transaction	62	Restricted Card
20	Invalid Response	63	Security Violation
21	No Action Taken	64	Original Amount Incorrect
22	Suspected Malfunction	65	Exceeds Withdrawal Frequency Limit
23	Unacceptable Transaction Fee	66	Card Acceptor Call Acquirer Security
24	File Update not Supported by Receiver	67	Hard Capture—Pick Up Card at ATM
25	Unable to Locate Record on File	68	Response Received Too Late
26	Duplicate File Update Record	75	Allowable PIN Tries Exceeded
27	File Update Field Edit Error	86	ATM Malfunction
28	File Update File Locked Out	87	No Envelope Inserted
29	File Update not Successful	88	Unable to Dispense
30	Format Error	89	Administration Error
31	Bank not Supported by Switch	90	Cut-off in Progress
32	Completed Partially	91	Issuer or Switch is Inoperative
33	Expired Card—Pick Up	92	Financial Institution not Found
34	Suspected Fraud—Pick Up	93	Trans Cannot be Completed
35	Contact Acquirer—Pick Up	94	Duplicate Transmission
36	Restricted Card—Pick Up	95	Reconcile Error
37	Call Acquirer Security—Pick Up	96	System Malfunction
38	Allowable PIN Tries Exceeded	97	Reconciliation Totals Reset
39	No CREDIT Account	98	MAC Error
40	Requested Function not Supported	99	Reserved for National Use

8.5 IP Payments Declined Codes

The following table presents the potential declined codes that may be presented in a declined transaction result:

Code	Response Text
100	System Exception
101	Invalid company identifier
102	Invalid account identifier
103	Invalid API username or password
104	Invalid transaction type identifier
105	Invalid channel identifier
106	Invalid currency identifier
107	Invalid transaction amount
108	No customer identifier supplied
109	No customer reference supplied
110	Invalid credit card number
111	Invalid credit card expiry date
112	Invalid source account number
113	Invalid source account routing number
114	Invalid escrow account number
115	Invalid escrow account routing number
116	Invalid destination account number
117	Invalid destination account routing number
118	Invalid customer identifier
119	Customer status not active
120	Account status not active
121	Account does not have any risk profile rules assigned
122	Registered customer details not found
123	Duplicate in document list
124	CVN required but not supplied
130	<PreviousTrnReceipt> has been supplied, but it requires a credit card capture, refund or cancel transaction type
150	Account not set up to accept supplied currency transactions
151	Account not set up correctly to accept supplied currency transactions
152	Account not set up to accept credit card transactions for the supplied credit card type
153	Account not set up to accept credit card transactions for the supplied amount
154	Merchant account details not set up correctly
155	Interface details not set up correctly
156	Auth transaction not found
157	Auth transaction was declined
158	Capture amount exceeds original auth plus any previous capture total
159	Purchase or Capture transaction was not found
160	Purchase or Capture transaction was declined
161	Refund amount exceeds original purchase or capture plus any previous refund total
162	Risk profile rules failed
170	Account not set up to accept direct entry transactions
171	Account not set up correctly to accept direct entry transactions
180	Exception encountered when retrieving the receipt number
181	Exception encountered when receiving transaction data from client

182	Exception encountered when creating transaction XML log
183	Exception parsing transaction XML
184	Exception validating transaction XML
190	Exception encountered when finding transaction identifier
191	Exception encountered when finding credit card interface to use
192	Exception encountered when submitting transaction to interface
193	Exception encountered when finding direct entry details
200	Interface error
201	Interface Error with successful automatic reversal
300	DE Dishonour - 01 Invalid BSB Number
301	DE Dishonour - 02 Payment Stopped
302	DE Dishonour - 03 Account Closed
303	DE Dishonour - 05 Invalid Account Number
304	DE Dishonour - 06 Refer to Customer
305	DE Dishonour - 07 Challenge Authority to Process
306	DE Dishonour - 09 Technically Invalid
307	DE Dishonour - 04 Account Holder Deceased
308	DE Manually Refunded
400	CC Chargeback - Documentation not Supplied
401	CC Chargeback - Documentation supplied was not legible
402	CC Chargeback - Signature supplied does not match signature on file
403	CC Chargeback - Transaction duplicated
404	CC Chargeback - Goods not delivered
500	Batch Record Exception
600	CC Manually Refunded
700	Invalid Disbursement XML
701	Disbursement XML amount total does not match the transaction amount
702	Account number must be supplied for each disbursement
703	Account number supplied for disbursement does not exist or account is inactive
704	Account number supplied for disbursement appears two or more times
997	Remote Interface Exception
998	Transaction Payment Cancelled
999	Timeout when waiting for a response