TNSPay Payment Client Interface Specification

Version 3.0.12

Transaction Network Services Ltd.,
Dunluce House,
East Point, Business Park
Dublin 3,
Republic of Ireland

Company confidential: The information contained within this documentation is subject to change without notification and was accurate at the time of print. Its contents are guidelines and are not intended to have any legal standing. Transaction Network Services (UK) Limited gives no undertaking that the contents are accurate and will not be held responsible for any losses, damages or costs incurred by following them. Information in sections is reliant upon outside agencies and may also change without notification to Transaction Network Services (UK) Limited. The use of Transaction Network Services (UK) Limited products and services are subject to our standard terms and conditions. The various product and company names found herein are trademarks of their respective companies.

Page 3 of 50

Section 1 Introduction

1.1 Purpose and Scope

The main purpose of this document is to provide a detailed interface specification of the TNSPay Payment Client Workstation to integrators and developers wishing to integrate with TNSPay Payment Client.

This document does not cover installation and administration of the TNS Payment Client; this is covered in the TNSPay Payment Client Administration Guide.

1.2 Abbreviations

Abbreviation	Description
Client	3rd Party software integrating with TNSPay Payment Client.
TPRG	TNSPay Retail Gateway.
POS	See Client.
TPPC	TNSPay Payment Client.

Section 2 System Requirements

2.1 Operating Systems

Operating System	TPPC Mode
Windows XP (SP3)	Interactive Workstation. Silent Workstation.
Windows 7	Interactive Workstation. Silent Workstation.
Windows 2003	Server. Proxy Client.
Windows 2008	Server. Proxy Client.

TPPC is currently supported on 32 bit version of these operating systems and on 64 bit versions of Windows 7 and Windows 2008 Server. Installing or running TPPC WOW-64 32 bit compatibility mode is not supported.

2.2 Minimum Requirements

The following are the minimum system requirements for TNSPay Payment Client (TPPC). Please note that although TPPC is supported on the following platforms, there is a possibility of hardware restrictions with the devices used. Please contact your TNS account manager if you have any questions regarding this issue.

To successfully run TPPC, you will require the following minimum system specification:

- An IBM or 100% compatible PC running one of the supported operating systems.
- A fully compatible display adapter and SVGA monitor capable of at least 800 x 600 resolution and 256 colours (1024 x768 and 16-bit colours recommended).
- 256Mb available RAM for Windows XP (512Mb or above recommended).
  512Mb available RAM for Windows 7, Windows 2003 Server and Windows 2008 Server (1024Mb or above recommended for all).
- A hard disk drive with at least 500Mb of free space (installation only, see Additional Requirements below for more details on disk space requirements).
- An IDE or SCSI CD-ROM drive (4x minimum).
- COM ports.
  1 dedicated COM port is required if a Chip and PIN device is to be installed.
  1 dedicated COM port is required if a slip printer is required.
- A receipt printer. Printers currently tested and supported are:
  EPSOM TM88 II and III (ESC-POS)
  EPSOM TM88 IV
  Star TSP600.
- TNSPay may also support additional ESC-POS, OPOS and Windows drivers but full

compatibility is not assured.

- The following software must be installed on the target computer:
  Microsoft .NET Framework Version 3.5 or 3.5 SP1
  Microsoft Windows Installer Version 3.1 or higher

2.3 Additional Requirements

Additional disk space is required to support the TNSPay Payment Client databases. This will vary depending on the volume of offline transactions that are likely to be stored, and the frequency of tasks that require an audit trail. Each database, subject to available disk space, has a maximum size of 4 GB. This equates to approximately 2 million records on the offline database.

TNS recommend that an internal housekeeping procedure be run regularly to ensure current backups exist in the event of a total system failure.

For details on installing, configuring and testing your TPPC software then please consult the relevant TNSPay Payment Client Manual(s).

Section 3 TNSPay Payment Client

This product has been developed to meet the demands of an ever changing card payment industry. In order to meet current and future market requirements, TNS has developed this product in line with payment industry best practices and industry pre-approval in mind. Deployment, functionality, and maintenance have all been simplified using the latest technologies and development principles.

To satisfy the variety of deployments in the card present market today, TPPC offers a number of configuration options. Configured as a Workstation, it resides with the Smart Card Reader and processing application and provides either a TCP or GUI interface.

TPPC Workstation captures information passed from the processing application (Client). This is passed to the card reader, prompting the customer to insert the correct card and any additional details, such as PIN number etc. This information is then processed by TPPC Workstation, including sending it on-line for authorisation if required. If Chip validation only is required, TPPC server passes the relevant data back to the client. In the event that the Chip is validated, but there is a negative or unsuccessful response from a third party the terminal can declare the transaction void.

Configured as a server, TPPC Server relays transaction requests received over TCP from any source, including a TPPC Workstation, to the TNSPay Multi Channel Payment Gateway performing validation and authorisation where required.

TPPC has a Cardholder Not Present GUI suitable for taking both cardholder present and card-not-present transactions. This is called the POSClient application.

Section 4 Getting Started

4.1 Overview

In order to integrate with TNSPay Payment Client, you need to develop a client application which is capable of transmitting data via TCP/IP.

To initiate an authorisation transaction, you must transmit an initial request via TCP/IP to the preconfigured TPPC listening port (default 30500).

If TPPC is to be configured to run in Server or Interactive Workstation mode, then your application simply needs to be able to receive a single response to the authorisation request.

If TPPC is to be configured to run in Silent Workstation mode, then your application must be able to manage multiple prompts and data requests at each step of the transaction. Typically in this mode, your application will be able to display operator prompts received from TPPC ("Insert Card", "Enter Cashback Amount" etc) from TPPC and respond to them if required. See Section 5.6 for information on when a response is required. Interactive Workstation may also require this capability if you intend using Card Aceptance or Surcharging.

It is also recommended that your application be able to manage the TPPC Keep Alive message. This is an ASCII 'ACK' message (HEX 0x41, 0x43, 0x4B) which is sent from TPPC to your application at a predetermined interval to advise you that it is still processing the transaction. This is particularly useful when TPPC is running in Interactive Workstation mode where there may be a significant delay before the client receives any response. The default interval is 15 seconds and is configurable.

4.2 Point to Point Encryption

TNSPay Payment Client supports Point to Point Encryption, sometimes referred to as End to End or Terminal Encryption in Workstation mode. With this feature enabled, cardholder data is encrypted by the EMV terminal prior to transmission to TNSPay Payment Client. While this provides additional security benefits, it also means that it is **never** possible for TNSPay Payment Client to display cardholder data. Whether or not you choose to enable Point to Point Encryption will depend on your business needs, and should be discussed with your TNS Account Manager.

Point to Point Encryption does not impact on the *structure* of any of the transaction processes outlined in this document. As Point to Point Encryption is *format preserving* (i.e. the format of the encrypted data is not changed) cardholder data will not on the face of it appear to be encrypted.

When operating in a Point to Point Encryption environment, **all** card data must be entered via the EMV terminal. See Section 5.7.13 for details on keying card details via the EMV Terminal.

4.3 Pennies For Charity

Pennies gives customers the option to donate a few pence to charity when paying for goods or services by card. It's just one press of a button or a click of the mouse and the job's done.

Please see the Adminsirtation Guide for more information on Pennies For Charity.

4.4 Transaction Process

The ultimate aim of integration with TPPC is to process card payment transactions.

Depending on the TPPC configuration, some or all of the following stages must be handled.

1. Transaction Initialisation Request
   The client must be able to generate a Flexible Format request containing the minimum tags required for the transaction type, and sending the request to TPPC.

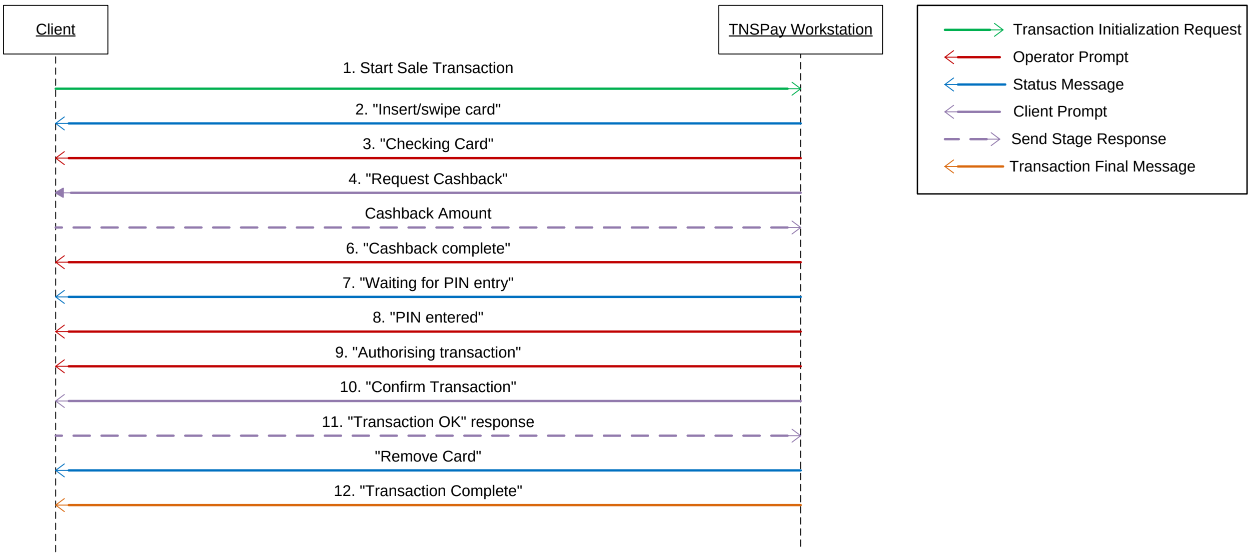
2. Receive Stage Request
   The client must be capable of receiving unsolicited requests from TPPC and processing them. This may be one of three distinct message types:
   a. Status Message - No response required from the client and no operator action required.
   b. Operator Prompt - No response required from the client however operator action is required (e.g. Insert Card)
   c. Client Prompt - Response is required from the client and operator action may be required (e.g. Enter Cashback Amount)

3. Send Stage Response
   Sent by the client to TPPC in response to a stage request that requires a client response.

4. Transaction Final Message
   The client must be capable of understanding when a final response has been received, and the final outcome of the transaction.

The following sequence diagram demonstrates a hypothetical transaction process, identifying the stages outlined above.

Fig. 4.1 Transaction Process Sequence Diagram.


Client
TNSPay Workstation
1. Start Sale Transaction
2. "Insert/swipe card"
3. "Checking Card"
4. "Request Cashback"
Cashback Amount
6. "Cashback complete"
7. "Waiting for PIN entry"
8. "PIN entered"
9. "Authorising transaction"
10. "Confirm Transaction"
11. "Transaction OK" response
"Remove Card"
12. "Transaction Complete"
Transaction Initialization Request
Operator Prompt
Status Message
Client Prompt
Send Stage Response
Transaction Final Message

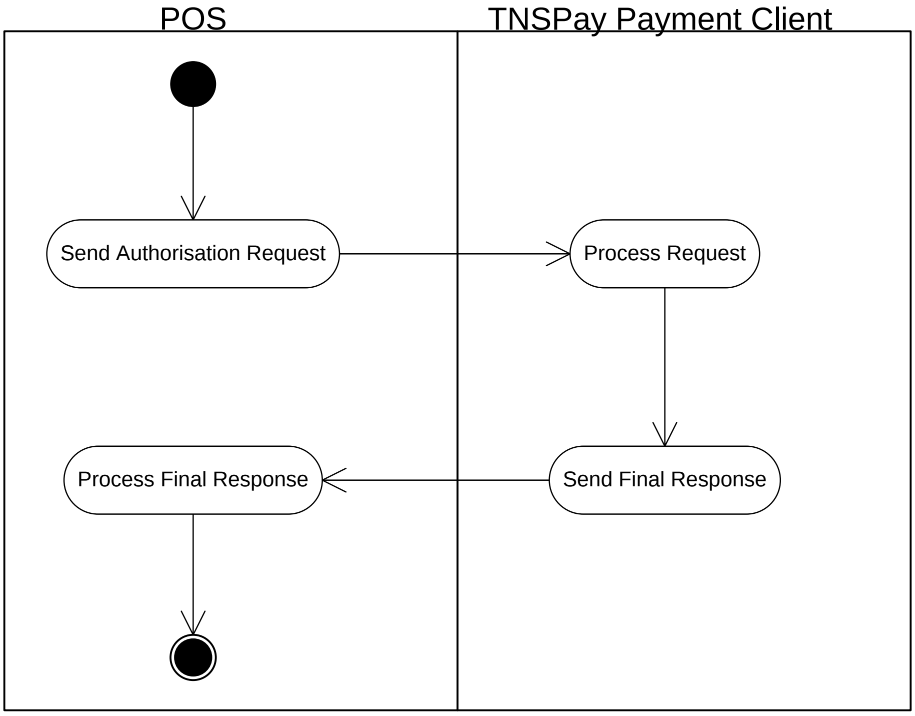
See Section 5 for detailed transaction examples.

4.4.1 Server and Interactive Workstation Transaction Process

When integrating with TPPC in Interactive Workstation mode, your job is pretty straightforward. An initial authorisation request message is submitted by your client to TPPC. Thereafter, the transaction is completely controlled by TPPC until the authorisation has been completed. TPPC will display all necessary user interaction through its own screens. Once the transaction has been completed, TPPC will send an authorisation response back to the client, and terminate the socket connection.

When interacting with TPPC in Server mode, things are equally straightforward. A single TCP request is submitted to TPPC which will return a single response once authorisation has been completed.

Fig. 4.2 TNSPay Payment Client State Diagram.



Section 5 contains detailed instructions on communicating with TPPC; including the TNS Flexible Format message protocol and examples of typical transactions and scenarios.

4.4.2 Silent Workstation Transaction Process

When integrating with TPPC in Silent Workstation mode, things are a little more complex. Your client is required to manage all user interaction and must be capable of processing numerous requests and information messages from TPPC before authorisation can be completed.

For example, for an ICC Sale Authorisation request, the card may support cash back. In this case, TPPC will send a response back to the client requesting a cash back amount. It will be up to your application to process this request, and interact with the user to allow a valid response to be returned to TPPC.

Equally, at various stages in the transaction process, it will be desirable to display status messages to the user. TPPC will send status messages to the client indicating the current status of the transaction. It is the responsibility of the client to interpret and display these messages correctly.

See Section 5 for details on communicating with TPPC. Section 5.5.2 contains specific information on how to identify specific requests from TPPC in Silent Workstation mode.

4.5 Device Integration

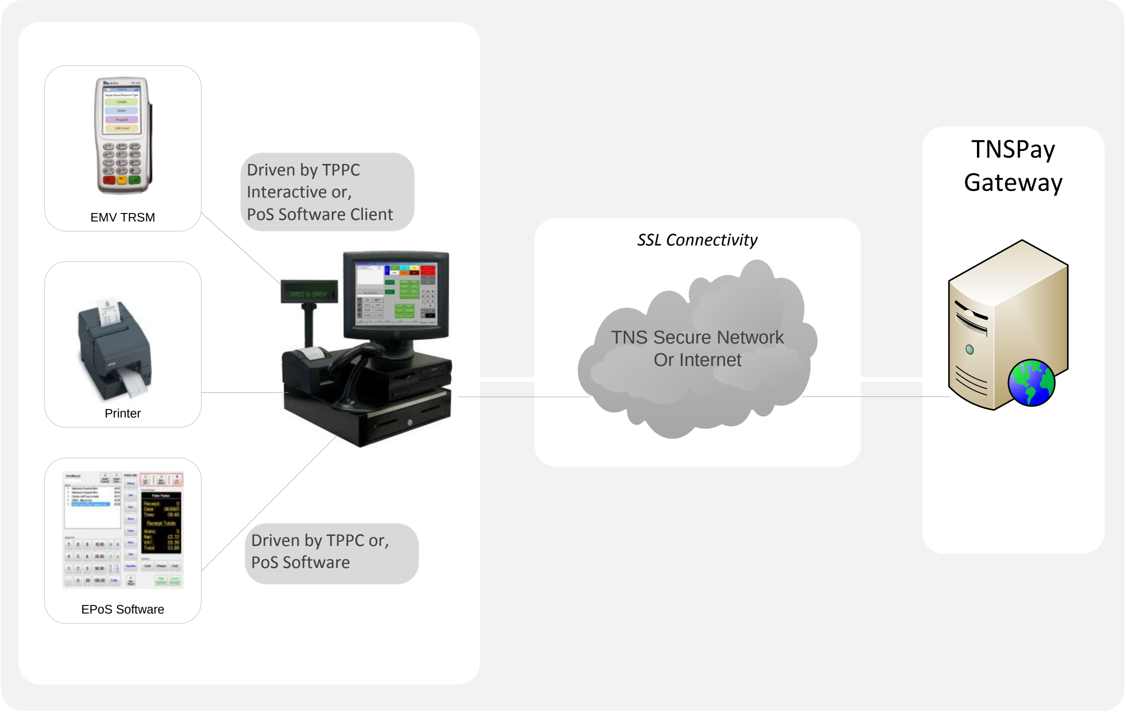
In a Cardholder Present environment, it is usual to interface with at least two external devices, a printer and a Card Reader/PIN Entry Device (PED).

TPPC supports multiple printer types, See Section 2 System Requirements in this document for more details.

TPPC supports Verifone Vx810 & Vx820 PIN Entry Devices.

Although TPPC will always directly drive the PED, you may determine whether or not TPPC will directly control the printer. If you choose to manage receipt printing from your own software, TPPC may be configured to provide an XML receipt schema, which is delivered over TCP. Section 7 of this document contains information detailed information on managing your own receipt printing process.

Fig. 4.3 Device Integration Overview.


EMV TRSM
Driven by TPPC Interactive or, PoS Software Client
Printer
EPoS Software
Driven by TPPC or, PoS Software
SSL Connectivity
TNS Secure Network Or Internet
TNSPay Gateway

Section 5 Communicating with TNSPay

5.1 Connecting to TNSPay Payment Client

Transaction data is sent by the client application over a TCP/IP port (30500 by default). The client application is responsible for opening the port and sending the initial transaction request. Once the initial request data has been sent from the front-end application the transaction process will start. Once the transaction is complete the response data will be returned to the front-end system over the same port.

5.2 Securing Communications

TPPC facilitates more secure communications over TCP/IP by means of a PKI interface. Use of TPPC PKI for communications between your software and TPPC is optional.

Using PKI, the client application generates a Public\Private Key Pair (RSA 2048 bit) and sends the Public key to TPPC. TPPC generates a strong symmetric key (AES 256 bit) which it encrypts using the RSA Public key. If also generates a Session Identifier. TPPC returns both the encrypted symmetric key and the Session Identifier to the client. The client then encrypts all further communications using the symmetric key.

To assist integrators in enabling this facility, a COM enabled .NET component (COMPlus) which provides both the initial PKI engine, the protocol and encryption engines for subsequent encrypted communications. See Section 9 COMPlus for details on COMPlus and the PKI solution protocol.

5.2.1 Transaction Flow

Once TPPC has received a transaction request (which will include details such as card number, transaction value etc.), it may communicate with the locally connected Chip and PIN device, as well as send the transaction on to the acquiring bank for authorisation.

Once the acquirer has responded, or TPPC has made a local decision, TPPC will generate a receipt on the preconfigured printer (if required), and return the authorisation response data to the client. The reply will contain the relevant response codes and a short descriptive message.

Fig. 5.1 TNSPay Payment Client Data Flow.

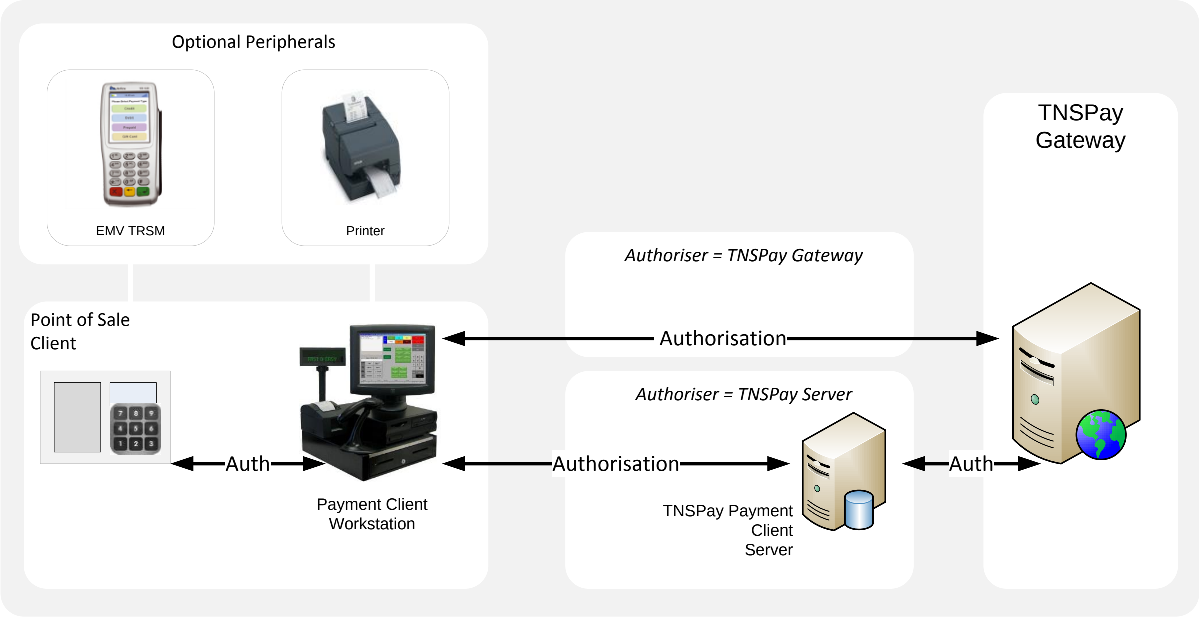

Optional Peripherals
EMV TRSM
Printer
TNSPay Gateway
Authoriser = TNSPay Gateway
Authorisation
Point of Sale Client
Auth
Payment Client Workstation
Authoriser = TNSPay Server
Authorisation
TNSPay Payment Client Server
Auth

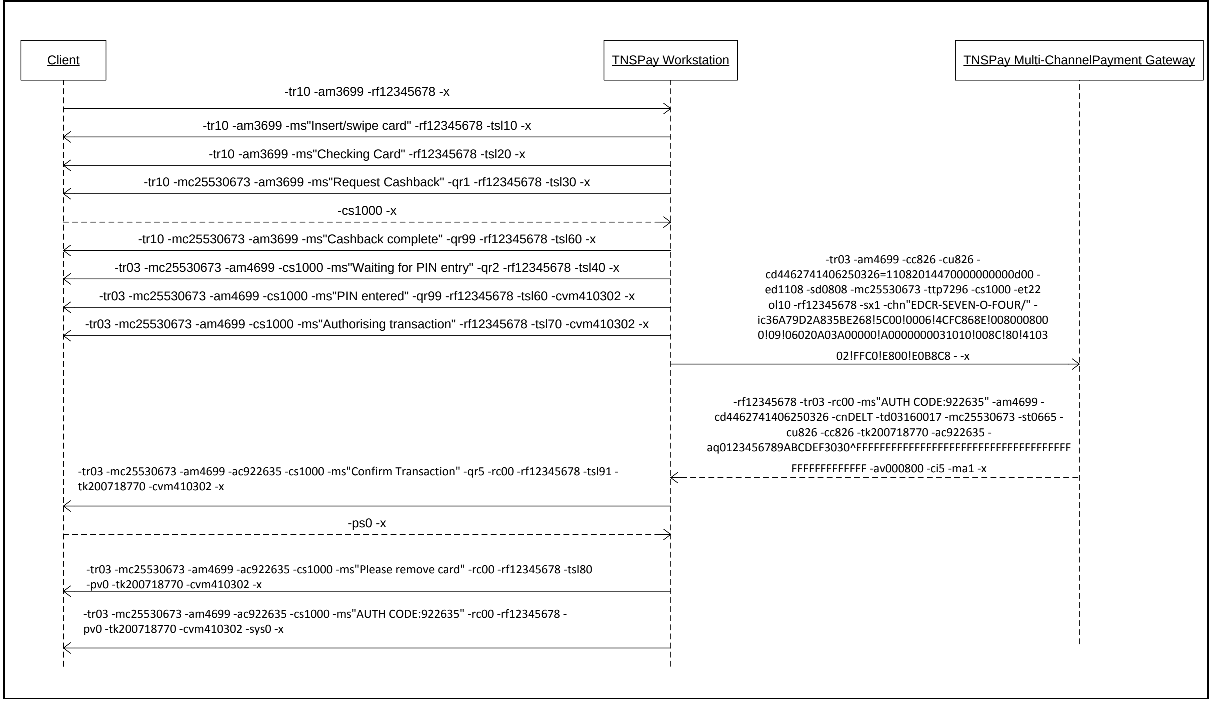
Table 5.1 below outlines the Transaction Flow for an ICC Sale with Cashback transaction. It demonstrates the data flow between the Client, TPPC and TPRG. The TPPC ↔ TPRG data flows are shown for illustration and completeness purposes only and you as an integrator will not be required to interact with these flows.

TNSPay Silent Workstation	
TNSPay ← Client	-tr10 -am3699 -rf12345678 -x
TNSPay → Client	-tr10 -am3699 -ms"Insert/swipe card" -rf12345678 -tsl10 -x
TNSPay → Client	-tr10 -am3699 -ms"Checking Card" -rf12345678 -tsl20 -x
TNSPay → Client	-tr10 -mc25530673 -am3699 -ms"Request Cashback" -qr1 -rf12345678 -tsl30 -x
TNSPay ← Client	-cs1000 -x
TNSPay → Client	-tr10 -mc25530673 -am3699 -ms"Cashback complete" -qr99 -rf12345678 -tsl60 -x
TNSPay → Client	-tr03 -mc25530673 -am4699 -cs1000 -ms"Waiting for PIN entry" -qr2 -rf12345678 -tsl40 -x
TNSPay → Client	-tr03 -mc25530673 -am4699 -cs1000 -ms"PIN stage complete" -qr99 -rf12345678 -tsl60 -cvm410302 -x
TNSPay → Client	-tr03 -mc25530673 -am4699 -cs1000 -ms"Authorising transaction" -rf12345678 -tsl70 -cvm410302 -x
TPPC → TPRG	-tr03 -am4699 -cc826 -cu826 -cd4462741406250326=11082014470000000000d00 -ed1108 -sd0808 -mc25530673 -ttp7296 -cs1000 -et22 ol10 -rf12345678 -sx1 -chn"EDCR-SEVEN-O-FOUR/" -ic36A79D2A835BE268!5C00!0006!4CFC868E!0080008000!09!06020A03A00000!A0000000031010!008C!80!410302!FFC0!E800!E0B8C8 - -x
TPPC ← TPRG	-rf12345678 -tr03 -rc00 -ms"AUTH CODE:922635" -am4699 -cd4462741406250326 -cnDELT -td03160017 -mc25530673 -st0665 -cu826 -cc826 -tk200718770 -ac922635 -aq0123456789ABCDEF3030^FFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFF -av000800 -ci5 -ma1 -x
TNSPay → Client	-tr03 -mc25530673 -am4699 -ac922635 -cs1000 -ms"Confirm Transaction" -qr5 -rc00 -rf12345678 -tsl91 -tk200718770 -cvm410302 -x
TNSPay ← Client	-ps0 -x
TNSPay → Client	-tr03 -mc25530673 -am4699 -ac922635 -cs1000 -ms"Please remove card" -rc00 -rf12345678 -tsl80 -pv0 -tk200718770 -cvm410302 -x
TNSPay → Client	-tr03 -mc25530673 -am4699 -ac922635 -cs1000 -ms"AUTH CODE:922635" -rc00 -rf12345678 -pv0 -tk200718770 -cvm410302 -sys0 -x
TNSPay Interactive Workstation	
TNSPay ← Client	-tr10 -am3699 -rf12345678 -x
TPPC → TPRG	-tr03 -am4699 -cc826 -cu826 -cd4462741406250326=11082014470000000000d00 -ed1108 -sd0808 -mc25530673 -ttp7296 -cs1000 -et22 ol10 -rf12345678 -sx1 -chn"EDCR-SEVEN-O-FOUR/" -ic36A79D2A835BE268!5C00!0006!4CFC868E!0080008000!09!06020A03A00000!A0000000031010!008C!80!410302!FFC0!E800!E0B8C8 - -x

TPPC ← TPRG	-rf12345678 -tr03 -rc00 -ms"AUTH CODE:922635" -am4699 -cd4462741406250326 -cnDELT -td03160017 -mc25530673 -st0665 -cu826 -cc826 -tk200718770 -ac922635 -aq0123456789ABCDEF3030^FFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFF -av000800 -ci5 -ma1 -x
TNSPay → Client	-tr03 -mc25530673 -am4699 -ac922635 -cs1000 -ms"AUTH CODE:922635" -rc00 -rf12345678 -pv0 -tk200718770 -cvm410302 -sys0 -x

This flow is further illustrated in the Fig 5.2 Sequence Diagram below.

Fig. 5.2 Silent Client ICC Sale With Cashback Sequence.



-tr10 -am3699 -rf12345678 -x

-tr10 -am3699 -ms"Insert/swipe card" -rf12345678 -tsl10 -x

-tr10 -am3699 -ms"Checking Card" -rf12345678 -tsl20 -x

-tr10 -mc25530673 -am3699 -ms"Request Cashback" -qr1 -rf12345678 -tsl30 -x

-cs1000 -x

-tr10 -mc25530673 -am3699 -ms"Cashback complete" -qr99 -rf12345678 -tsl60 -x

-tr03 -mc25530673 -am4699 -cs1000 -ms"Waiting for PIN entry" -qr2 -rf12345678 -tsl40 -x

-tr03 -mc25530673 -am4699 -cs1000 -ms"PIN entered" -qr99 -rf12345678 -tsl60 -cvm410302 -x

-tr03 -mc25530673 -am4699 -cs1000 -ms"Authorising transaction" -rf12345678 -tsl70 -cvm410302 -x

-tr03 -am4699 -cc826 -cu826 -cd4462741406250326=11082014470000000000d00 -ed1108 -sd0808 -mc25530673 -ttp7296 -cs1000 -et22 ol10 -rf12345678 -sx1 -chn"EDCR-SEVEN-O-FOUR/" -ic36A79D2A835BE268!5C00!0006!4CFC868E!0080008000!09!06020A03A00000!A0000000031010!008C!80!410302!FFC0!E800!E0B8C8 - -x

-rf12345678 -tr03 -rc00 -ms"AUTH CODE:922635" -am4699 -cd4462741406250326 -cnDELT -td03160017 -mc25530673 -st0665 -cu826 -cc826 -tk200718770 -ac922635 -aq0123456789ABCDEF3030^FFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFF -av000800 -ci5 -ma1 -x

-tr03 -mc25530673 -am4699 -ac922635 -cs1000 -ms"Confirm Transaction" -qr5 -rc00 -rf12345678 -tsl91 -tk200718770 -cvm410302 -x

-ps0 -x

-tr03 -mc25530673 -am4699 -ac922635 -cs1000 -ms"Please remove card" -rc00 -rf12345678 -tsl80 -pv0 -tk200718770 -cvm410302 -x

-tr03 -mc25530673 -am4699 -ac922635 -cs1000 -ms"AUTH CODE:922635" -rc00 -rf12345678 -pv0 -tk200718770 -cvm410302 -sys0 -x

5.3 Monitoring Connection Status

When operating in Interactive Workstation Mode the socket connection could be held open for a considerable period whilst the chip and pin transaction takes place. In Interactive Workstation mode, TPPC will send an 'ACK' message (HEX 0x41, 0x43, 0x4B) back to the client over the open socket connection. The default interval time for these is every 15 seconds. This interval may be configured using the TPPC Admin application. See the TNSPay Payment Client Administration Guide for more information on configuring this option.

It is recommended that if the client does not receive an 'ACK' response within 20 seconds, the front-end should disconnect its socket connection and re-establish the link. At this stage the front-end should pass an exact duplicate of the original request with an additional flag -ryt (for retry transaction). If an invalid Retry request is received an -rc30 response will be returned by TPPC.

5.4 TNS FlexibleFormat Mnemonic Protocol

TPPC uses the standard TNS Flexible Format Mnemonic protocol for processing transactions. All transactions submitted to TPPC must confirm to the TNS Flexible Format specification.

The following sections provide examples of Flexible Format message exchanges typically seen when integrating with TPPC. The examples only show the TCP message content between the client and TPPC Workstation. For TPPC, this may constitute a series of exchanges between client and TPPC. For TPPC, this will typically consist of only one message pair. See the TNSPay Payment Client User Guide for more detail on the transaction process for the Interactive Workstation Mode.

5.5 TNSPay Workstation Modes

TNSPay Workstation can be run in two distinct modes: Interactive or Silent.

5.5.1 Interactive Mode

In Interactive Mode, TPPC takes control of the transaction and displays its own Graphical User Interface (GUI) to the operator.

Once the initial request is sent, the client will receive no further TCP communication from TPPC until the transaction is complete.

See the TNSPay Payment Client User Guide for a detailed description of the Interactive Mode GUI.

5.5.2 Silent Mode

In Silent Mode, all user-interaction and GUI management is under control of the 3rd party utilising the TPPC Workstation, which in turn keeps the client up-to-date with the transaction state by regularly sending it status messages. Certain messages to the client may also be a request for information (e.g. cashback amount).

Messages to the 3rd-party client usually contain a status message that should be displayed to the operator. The transaction-state associated with each message is identified with unique *transaction-state-level* value (table 5.2). This is provided so that integrators may customize their payment-interface screens for different transaction states, as opposed to merely displaying the status message received.

Value	Description
10	Waiting for card.
11	Waiting for key-card details on terminal.
20	Card inserted/removed.
30	Request Cashback.
40	Waiting for PIN Entry (terminal).
41	Waiting for application selection (terminal).
50	Pin-bypass timeout.
60	Pin Entered.
70	Authorizing transaction.
71	Printing.
80	Please Remove Card.
80	Waiting for referral code.
91	Confirm Transaction.

5.5.2.1 Silent Client Transaction Processing

Clients integrating with TNSPay Payment Client in the Silent-Client Workstation mode should ideally develop their interface not to be *stateful*; i.e. other than displaying status messages, action only need be taken in response to messages received that specify the *Query Response* (-qr) mnemonic.

Query Response messages indicate that an action may be required from either the operator or the cardholder (e.g. Enter Cashback Amount, Insert Card). Transaction flow may be paused until a response is received by TPPC.

First Digit	Description
1	Waiting for cash-back.
2	Enter PIN or bypass.
3	PIN Bypass timeout.
4	Waiting for referral.
5	Confirm transaction.
6	Accept/Decline card-type
10	Apply Surcharge
11	Waiting for key-card-details on EMV (VSP only).
99	Clear the previous prompt/Transaction Stage Complete.

Note: A transaction can be considered **complete** if a message is received that contains a response code, but no TSL mnemonic. The success or failure of the transaction may be determined from the response code mnemonic (-rc), where successful response codes include '00', '08', '11', 'Y1', 'Y3'.

5.6 Response Templates

TPPC provides 3 different response templates, Low(Default), Medium and High. Each template contains progressively more mnemonic tags. The Response Template controls the amount of information that is returned to the client in any response message. Only the tags relevant to a particular response will be returned. The response template is set using the TPPC Admin application.

Only the Low(Default) response template qualifies for automatic acquirer accreditation. You should contact your acquirer before enabling any other template.

Description	Low(Default)	Medium	High
Transaction authorisation code.	ac	ac	ac
Application identifier.			aid
Application interchange profile.			aip
Transaction amount.	am	am	am
Application label			apl
Application preferred name			apn
Icc response data			aq
Transaction cryptogram.			aqc
Application transaction counter.			atc
Application usage control.			auc
Address verification results.			av
Transaction country code.		cc	cc
Card details (number).		cd	cd
Cardholder name.		chn	chn
Customer instruction field.			ci
Cryptogram information data.			cid
Command result	cmr	cmr	cmr
Command result text	cmt	cmt	cmt
Card name.		cn	cn
Transaction cash amount.	cs	cs	cs
Cryptogram transaction type.			ctt
Transaction currency code.		cu	cu
Cardholder verification method list results.	cvm		cvm
Description 1.			ds1
Description 2.			ds2
Description 3.			ds3
Description 4.			ds4
Description 5.			ds5
Card expiry date.	ed	ed	ed
Emv terminal type.			et
House number.			hn
Issuer action codes.			iac

Issuer application data.			iad
Emv chip data forwarded from the front-end.		ic	ic
Card issue number.		is	is
Method of authorisation.			ma
Merchant number.	mc	mc	mc
Response message.	ms	ms	ms
Pos entry mode.			pem
Receipt print status.		ps	ps
Primary account number (pan) sequence number.			psn
Pin verification result	pv	pv	pv
Query response.	qr	qr	qr
Response code.	rc	rc	rc
Transaction reference number.	rf	rf	rf
Referral telephone number	rtn	rtn	rtn
Card start date.		sd	sd
Sub-response code	src	src	src
Transaction counter.		st	st
Swiped track 2 data.		sw2	sw2
System status	sys	sys	sys
Terminal capabilities.			tcb
Terminal id.	td	td	td
Transaction key.	tk	tk	tk
Token	tov	tov	tov
Transaction type.	tr	tr	tr
Tokenization sub-response code	trc	trc	trc
Transaction status information.			tsi
Transaction state level.	tsl	tsl	tsl
Terminal type.			ttp
Terminal verification results.			tvr
Contactless transaction outcome	txno	txno	txno
Unique transaction identifier	uid	uid	uid
Unpredictable number.			upn
Application version number.			ver
Post code.			zp
Donation amount	don	don	don
Pennies merchant number	pom	pom	pom
Pennies store number	pos	pos	pos

5.7 Examples

This section provides examples of message content for various types of transactions.

As the content of messages varies greatly depending on the transaction type and system configuration and deployment, these examples are not exhaustive. Please refer to the TNS FlexibleFormat Mnemonic Reference Guide for a more comprehensive definition of individual Flexible Format tags.

In all request examples, it is assumed that TPPC Workstation is managing Receipt Printing, and that all print attempts are successful. See Section 7 for details on receipt generation options.

To fully illustrate the typical available response data, all response examples assume that the Response Template is set to Default.

5.7.1 Request Data

Only 3 populated Flexible Format tags are required to initiate a Cardholder Present authorisation request, although further additional tags may be included. *(M = Mandatory, O = Optional)*

Request Fields			
Tag	**Description**	**Required**	**Example**
rf	Transaction Reference	M	-rfeos00074
am	Transaction Amount	M	-am5000
tr	Transaction Type	M	-tr10
cu	Transaction Currency	O	-cu826
mc	Merchant ID	O	-mc25530673
et	EMV Terminal Type	O	-et22
ca	Card Acceptance Flag Buffer	O	-ca0BFA00
td	Terminal ID	O	-td03160017
cc	Country Code	O	-826
-ds*n*	Description Field (where *n* = 1-5)	O	-ds1"EMV Sale"

Request Fields			
Tag	**Description**	**Required**	**Example**
rf	Transaction Reference	M	-rfeos00074
am	Transaction Amount	M	-am5000
tr	Transaction Type	M	-tr10
cd	Card Details	M	-cd4444333322221111
ed	Expiration Date	M	-ed0912
sc	Card Security Code	M	-sc123
cu	Transaction Currency	O	-cu826
mc	Merchant ID	O	-mc25530673

ttp	Terminal Type	O	-ttp7296
hn	House Number	O	-hn"22 Drury Street"
zp	Post Code	O	-zp"KT9 2NY"
td	Terminal ID	O	-td03160017
cc	Country Code	O	-826
ds*n*	Description Field (where *n* = 1-5)	O	-ds1"EMV Sale"

5.7.2 Example: ICC Sale with Cashback Enabled

This example demonstrates the message exchange between TPPC and a client application for a Sale request on an ICC card. The card used is a Delta card, which permits Sale with Cashback. TPPC is also configured to process Sale with Cashback transactions.

The request provides only the mandatory tags identified in Table Section 5.7.1, and leaves it to TPPC to fill all other authorisation data.

TNSPay Silent Workstation	
TNSPay ← Client	-tr10 -am3699 -rf12345678 -x
TNSPay → Client	-tr10 -am3699 -ms"Insert/swipe card" -rf12345678 -tsl10 -x
TNSPay → Client	-tr10 -am3699 -ms"Checking Card" -rf12345678 -tsl20 -x
TNSPay → Client	-tr10 -mc25530673 -am3699 -ms"Request Cashback" -qr1 -rf12345678 -tsl30 -x
TNSPay ← Client	-cs1000 -x
TNSPay → Client	-tr10 -mc25530673 -am3699 -ms"Cashback complete" -qr99 -rf12345678 -tsl60 -x
TNSPay → Client	-tr03 -mc25530673 -am4699 -cs1000 -ms"Waiting for PIN entry" -qr2 -rf12345678 -tsl40 -x
TNSPay → Client	-tr03 -mc25530673 -am4699 -cs1000 -ms"PIN stage complete" -qr99 -rf12345678 -tsl60 -cvm410302 -x
TNSPay → Client	-tr03 -mc25530673 -am4699 -cs1000 -ms"Authorising transaction" -rf12345678 -tsl70 -cvm410302 -x
TNSPay → Client	-tr03 -mc25530673 -am4699 -ac922635 -cs1000 -ms"Confirm Transaction" -qr5 -rc00 -rf12345678 -tsl91 -tk200718770 -cvm410302 -x
TNSPay ← Client	-ps0 -x
TNSPay → Client	-tr03 -mc25530673 -am4699 -ac922635 -cs1000 -ms"Please remove card" -rc00 -rf12345678 -tsl80 -pv0 -tk200718770 -cvm410302 -x
TNSPay → Client	-tr03 -mc25530673 -am4699 -ed1301 -ac922635 -cs1000 -ms"AUTH CODE:922635" -rc00 -rf12345678 -pv0 -tk200718770 -cvm410302 -sys0 –x
TNSPay Interactive Workstation	
TNSPay ← Client	-tr10 -am3699 -rf12345678 –x
TNSPay → Client	-tr03 -mc25530673 -ed1301 -am4699 -ac922635 -cs1000 -ms"AUTH CODE:922635" -rc00 -rf12345678 -pv0 -tk200718770 -cvm410302 -sys0 -x

5.7.3 Example: ICC Sale with Cashback Disabled

This example demonstrates the message exchange between TNSPay and a client application for a more comprehensive Sale request on an ICC card. The card used is a Delta card, which permits Sale with Cashback, however TNSPay is now configured not to process Sale with Cashback transactions, so no request for a cashback amount is transmitted.

Table 5.11 ICC Sale With No Cashback Example.

TNSPay Silent Workstation	
TNSPay ← Client	-ca000000 -rfIC123456 -td03160017 -tr10 -am100 -cc826 -cu826 -ds1"EMV Transaction" -ds2"ICC Request" -mc25530673 -x
TNSPay → Client	-tr10 -mc25530673 -am100 -ms"Insert/swipe card" -rfIC123456 -tsl10 -x

TNSPay → Client	-tr10 -am100 -mc25530673 -rfIC123456 -tsl20 -ms"Checking Card" -x
TNSPay → Client	-tr10 -am100 -mc25530673 -rfIC123456 -tsl40 -qr2 -ms"Waiting for PIN entry" -x
TNSPay → Client	-tr10 -am100 -cvm410302 -mc25530673 -rfIC123456 -tsl60 -qr99 -ms"PIN stage complete" -x
TNSPay → Client	-tr10 -am100 -cvm410302 -mc25530673 -rfIC123456 -tsl70 -ms"Authorising transaction" -x
TNSPay → Client	-tr10 -rc00 -am100 -tk200718780 -ac295182 -cvm410302 -mc25530673 -ms"Confirm Transaction" -rfIC123456 -tsl91 -qr5 -x
TNSPay ← Client	-ps0 -x
TNSPay → Client	-tr10 -mc25530673 -am100 -ac295182 -ms"Please remove card" -rc00 -rfIC123456 -tsl80 -pv0 -tk200718780 -cvm410302 -x
TNSPay ← Client	-tr10 -mc25530673 -am100 -ac295182 -ed1301 -ms"AUTH CODE:295182" -rc00 -rfIC123456 -pv0 -tk200718780 -cvm410302 -sys0 –x
TNSPay Interactive Workstation	
TNSPay ← Client	-ca000000 -rfIC123456 -td03160017 -tr10 -am100 -cc826 -cu826 -ds1"EMV Transaction" -ds2"ICC Request" -mc25530673 -x
TNSPay → Client	-tr10 -mc25530673 -am100 -ac295182 -ms"AUTH CODE:295182" -ed1301 -rc00 -rfIC123456 -pv0 -tk200718780 -cvm410302 -sys0 -x

5.7.4 Example: ICC Fallback to Swipe Sale with Cashback

This example demonstrates the message exchange between TNSPay and a client application for a Sale request. The ICC read fails, resulting in a fallback to a swipe read of the card. The card used is a Visa Debit card, which permits Sale with Cashback. TNSPay is also configured to process Sale with Cashback transactions.

The request provides only the mandatory tags identified in Table Section 5.7.1, and leaves it to TNSPay to fill all other authorisation data.

Table 5.12 ICC Fallback to Swipe Sale With Cashback Example.

TNSPay Silent Workstation	
TNSPay ← Client	-tr10 -am3699 -rf12345678 -x
TNSPay → Client	tr10 -am3699 -ms"Insert/swipe card" -rf12345678 -tsl10 -x
TNSPay → Client	-tr10 -am3699 -rf12345678 -tsl60 -qr99 -ms"Cannot read card" -x
TNSPay → Client	-tr10 -am3699 -ms"Please remove card" -rf12345678 -tsl80 -x
TNSPay → Client	tr10 -am3699 -ms"Insert/swipe card" -rf12345678 -tsl10 -x
TNSPay → Client	-tr10 -am3699 -rf12345678 -tsl60 -qr99 -ms"Cannot read card" -x
TNSPay → Client	-tr10 -am3699 -ms"Please remove card" -rf12345678 -tsl80 -x
TNSPay → Client	tr10 -am3699 -ms"Insert/swipe card" -rf12345678 -tsl10 -x
TNSPay → Client	-tr10 -am3699 -rf12345678 -tsl60 -qr99 -ms"Cannot read card" -x
TNSPay → Client	-tr10 -am3699 -ms"Please remove card" -rf12345678 -tsl80 -x
TNSPay → Client	-tr10 -am3699 -ms"Swipe Card" -rf12345678 -tsl10 -x
TNSPay → Client	-tr10 -am3699 -ms"Card Swiped" -rf12345678 -tsl20 -x
TNSPay → Client	-tr10 -mc25530673 -am3699 -ms"Request Cashback" -qr1 -rf12345678 -tsl30 -x
TNSPay ← Client	-cs1000 -x
TNSPay → Client	-tr10 -mc25530673 -am3699 -ms"Cashback complete" -qr99 -rf12345678 -tsl60 -x
TNSPay → Client	-tr03 -mc25530673 -am4699 -cs1000 -ms"Authorising transaction" -rf12345678 -tsl70 -x
TNSPay → Client	-tr03 -mc25530673 -am4699 -ac174586 -cs1000 -ms"Confirm Transaction" -qr5 -rc00 -rf12345678 -tsl91 -tk200718804 -x
TNSPay ← Client	-ps0 -x
TNSPay → Client	-tr03 -mc25530673 -am4699 -ed1301 -ac174586 -cs1000 -ms"AUTH CODE:174586" -rc00 -rf12345678 -pv1 -tk200718804 -sys0 -x
TNSPay Interactive Workstation	
TNSPay ← Client	-tr10 -am3699 -rf12345678 -x
TNSPay → Client	-tr03 -mc25530673 -am4699 -ac174586 -ed1301 -cs1000 -ms"AUTH CODE:174586" -rc00 -rf12345678 -pv1 -tk200718804 -sys0 -x

5.7.5 Example: Swipe Sale with Cashback Disabled

This example demonstrates the message exchange between TNSPay and a client application for a more comprehensive Sale request on a Swipe card. The card used is a Delta card, which permits Sale with Cashback; however TNSPay is now configured not to process Sale with Cashback transactions, so no request for a cashback amount is transmitted.

Table 5.13 Swipe Sale With No Cashback Example.

TNSPay Silent Workstation	
TNSPay ← Client	-tr10 -am100 -ca000000 -cc826 -cu826 -ds1"EMV Transaction" -ds2"ICC Request" -mc25530673 -rfIC123456 -td03160017 -x
TNSPay → Client	-tr10 -mc25530673 -am100 -ms"Insert/swipe card" -rfIC123456 -tsl10 -x
TNSPay → Client	-tr10 -mc25530673 -am100 -ms"Card Swiped" -rfIC123456 -tsl20 -x
TNSPay → Client	-tr10 -mc25530673 -am100 -ms"Authorising transaction" -rfIC123456 -tsl70 -x
TNSPay → Client	-tr10 -mc25530673 -am100 -ac364661 -ms"Confirm Transaction" -qr5 -rc00 -rfIC123456 -tsl91 -tk200718786 -x
TNSPay ← Client	-ps0 -x
TNSPay → Client	-tr10 -mc25530673 -am100 -ed1301 -ac364661 -ms"AUTH CODE: 364661 " -rc00 -rfIC123456 -pv1 -tk200718786 -sys0 -x
TNSPay Interactive Workstation	
TNSPay ← Client	-tr10 -am100 -ca000000 -cc826 -cu826 -ds1"EMV Transaction" -ds2"ICC Request" -mc25530673 -rfIC123456 -td03160017 -x
TNSPay → Client	-tr10 -mc25530673 -am100 -ac364661 -ms"AUTH CODE: 364661 " -ed1301 -rc00 -rfIC123456 -pv1 -tk200718786 -sys0 -x

5.7.6 Example: Swipe Sale Fallback to Keyed

This example demonstrates the message exchange between TNSPay and a client application for a Sale request. The Magnetic Stripe read fails, resulting in a fallback to allow the operator key the card details. The card used is a MasterCard Credit card, which does not permit Sale with Cashback.

This transaction is very different to the ICC Fallback to Swipe demonstrated in Section 5.7.4. In Silent Mode TPPC does not implicitly handle fallback from Swipe to Keyed, as TPPC has no way of detecting whether or not a card contains a chip (not permitted to fallback to keyed). TNS recommend that the client follows the same approach adopted by TPPC in Interactive Mode, and requests the operator to perform a visual check on the card before permitting a keyed cardholder present transaction.

The request provides only the mandatory tags identified in Table Section 5.7.1, and leaves it to TNSPay to fill all other authorisation data.

If Point to Point Encryption is enabled, you must use the method described in Section 5.7.13.

Table 5.14 Swipe Fallback to Keyed Sale Example.

TNSPay Silent Workstation	
TNSPay ← Client	-tr10 -am3699 -rf12345678 -x
TNSPay → Client	-tr10 -am3699 -ms"Insert/swipe card" -rf12345678 -tsl10 -x
TNSPay → Client	-tr10 -am3699 -ms"Swipe Card" -rf12345678 -tsl10 -x
TNSPay → Client	-tr10 -am3699 -ms"Swipe Card" -rf12345678 -tsl10 -x
TNSPay → Client	-tr10 -am3699 -ms"Cannot read card" -rc30 -rf12345678 -sys0 -x
TPPC has now finalised the transaction and returned a response code. TPPC does not require a confirmation response. The operator should now perform a visual check to ensure that the card does not contain a chip before attempting to key the transaction. The keyed transaction is treated as a new transaction by TPPC.	
TNSPay ← Client	-tr20 -cd4000000000000002 -ed1209 -am3699 -rf12345678 -x
TNSPay → Client	-tr20 -mc25530673 -am3699 -ms"Authorising transaction" -rf12345678 -tsl70 -x
TNSPay → Client	-tr20 -mc25530673 -am3699 -ac451966 -ms"Confirm Transaction" -qr5 -rc00 -rf12345678 -tsl91 -tk200718827 -x

TNSPay ← Client	-ps0 -x
TNSPay → Client	-tr20 -mc25530673 -am3699 -ac451966 -ms"AUTH CODE:451966" -rc00 -rf12345678 -pv1-ed1209 -tk200718827 -sys0 -x
TNSPay Interactive Workstation	
TNSPay ← Client	-tr10 -am3699 -rf12345678 -x
In Interactive Mode, TPPC handles the keyed fallback within the Interactive process, prompting the operator to perform a visual check to ensure that the card does not contain a chip before attempting to key the transaction. At the operator's request, TPPC will then initiate a new keyed transaction. It is important that the client, if it validates responses from TPPC, be able to handle the fact that the transaction type in the final response will be different to the transaction type originally submitted.	
TNSPay → Client	-tr20 -mc25530673 -am3699 -ac451966 -ms"AUTH CODE:451966" -rc00 -rf12345678 -pv1 -tk200718827 –ed1209 -sys0 -x

5.7.7 Example: ICC Refund

This example demonstrates the message exchange between TNSPay and a client application for a Refund request on an ICC card.

The request provides only the mandatory tags identified in Table Section 5.7.1, and leaves it to TNSPay to fill all other authorisation data.

Table 5.15 ICC Refund Example.

TNSPay Silent Workstation	
TNSPay ← Client	-tr58 -am3699 -rf12345678 -x
TNSPay → Client	-tr58 -am3699 -ms"Insert/swipe card" -rf12345678 -tsl10 -x
TNSPay → Client	-tr58 -am3699 -ms"Checking Card" -rf12345678 -tsl20 -x
TNSPay → Client	-tr61 -am3699 -mc25530673 -rf12345678 -tsl70 -ms"Authorising transaction" -x
TNSPay → Client	-tr61 -mc25530673 -am3699 -ac40580 -ms"Confirm Transaction" -qr5 -rc00 -rf12345678 -tsl91 -tk200718797 -x
TNSPay ← Client	-ps0 -x
TNSPay → Client	-tr61 -mc25530673 -am3699 -ac40580 -ms"Please remove card" -rc00 -rf12345678 -tsl80 -pv1 -tk200718797 -x
TNSPay → Client	-tr61 -mc25530673 -am3699 -ac40580 -ms"AUTH CODE: 40580" -rc00 -rf12345678 -pv1 -tk200718797 -ed1301 -sys0 -x
TNSPay Interactive Workstation	
TNSPay ← Client	-tr58 -am3699 -rf12345678 -x
TPPC→ TPRG	-tr61 -am3699 -cc826 -cu826 -cd4929498311400002 -ed1103 -mc25530673 -ttp7296 -et22 -ol10 -rf12345678 -sx1 -x
TPPC ← TPRG	-rf12345678 -tr61 -rc00 -ms"REFUND ACCEPTED" -st2209 -am3699 -cd4929498311400002 -ed1103 -cnVISA -mc25530673 -td03160017 -cu826 -cc826 -ac0 -ci9 -ma0 -tk200718797 -x
TNSPay → Client	-tr61 -mc25530673 -am3699 -ac40580 -ms"AUTH CODE: 40580" -rc00 -rf12345678 -pv1 -tk200718797-ed1301 -sys0 -x

5.7.8 Example: Swipe Refund

This example demonstrates the message exchange between TNSPay and a client application for a Refund request on a Swipe card.

The request provides only the mandatory tags identified in Table Section 5.7.1, and leaves it to TNSPay to fill all other authorisation data.

Table 5.16 Swipe Refund Example.

TNSPay Silent Workstation	
TNSPay ← Client	-tr58 -am3699 -rf12345678 -x
TNSPay → Client	-tr58 -am3699 -ms"Insert/swipe card" -rf12345678 -tsl10 -x
TNSPay → Client	-tr58 -am3699 -ms"Card Swiped" -rf12345678 -tsl20 -x
TNSPay → Client	-tr58 -mc25530673 -am3699 -ms"Authorising transaction" -rf12345678 -tsl70 -x
TPPC→ TPRG	-tr58 -am3699 -cc826 -cu826 -cd;5301250070000191=12091010912345678901? -ed1209 -mc25530673 -ttp7296 -et22 -ol10 -rf12345678 -sx1 -x
TPPC ← TPRG	-rf12345678 -tr58 -rc00 -ms"REFUND ACCEPTED" -st3642 -am3699 -cd5301250070000191 -ed1209 -cnMCRD -mc25530673 -td03160017 -cu826 -cc826 -ac0 -ci6 -ma0 -tk200718799 -x
TNSPay → Client	-tr58 -mc25530673 -am3699 -ac38500 -ms"Confirm Transaction" -qr5 -rc00 -rf12345678 -tsl91 -tk200718799 -x

TNSPay ← Client	-ps0 -x
TNSPay → Client	-tr58 -mc25530673 -am3699 -ac38500 -ms"AUTH CODE: 38500" -rc00 -rf12345678 -ed1301 -pv1 -tk200718799 -sys0 -x
TNSPay Interactive Workstation	
TNSPay ← Client	--tr58 -am3699 -rf12345678 -x
TPPC→ TPRG	-tr58 -am3699 -cc826 -cu826 -cd;5301250070000191=12091010912345678901? -ed1209 -mc25530673 -ttp7296 -et22 -ol10 -rf12345678 -sx1 -x
TPPC ← TPRG	-rf12345678 -tr58 -rc00 -ms"REFUND ACCEPTED" -st3642 -am3699 -cd5301250070000191 -ed1209 -cnMCRD -mc25530673 -td03160017 -cu826 -cc826 -ac0 -ci6 -ma0 -tk200718799 -x
TNSPay → Client	-tr58 -am3699 -cc826 -cu826 -cd;4539791001730106=12091010912345678901?< -ed1209 -mc25530673 -td03160017 -ttp7296 -rf12345678 -rc00 -cnDELT -ms"AUTH CODE: 37320" -ac37320 -ed1301 -ci5 -ma0 -st3336 -tk200401075 -pem11 -x

5.7.9 Example: ICC Sale Card Removed Prematurely

This example demonstrates the message exchange between TNSPay and a client application for a Sale request on an ICC Card. The card is prematurely removed from the device at the PIN Entry Stage resulting in a declined transaction.

The request provides only the mandatory tags identified in Table Section 5.7.1, and leaves it to TNSPay to fill all other authorisation data.

Table 5.17 ICC Sale Card Removed Prematurely Example.

TNSPay Silent Workstation	
TNSPay ← Client	-tr10 -am3699 -rf12345678 -x
TNSPay → Client	-tr10 -am3699 -ms"Insert/swipe card" -rf12345678 -tsl10 -x
TNSPay → Client	-tr10 -am3699 -ms"Checking Card" -rf12345678 -tsl20 -x
TNSPay → Client	-tr03 -mc25530673 -am3699 -ms"Waiting for PIN entry" -qr2 -rf12345678 -tsl40 -x
TNSPay → Client	-tr03 -mc25530673 -am3699 -ms"Card Removed" -qr99 -rc05 -rf12345678 -tsl60 -x
TNSPay → Client	-tr03 -mc25530673 -am3699 -ms"Confirm Transaction" -qr5 -rc05 -rf12345678 -tsl91 -x
TNSPay → Client	-ps0 -x
TNSPay → Client	-tr03 -mc25530673 -am3699 -ms"Card Removed" -rc05 -rf12345678 -ed1301 -sys0 -x
TNSPay Interactive Workstation	
TNSPay ← Client	-tr10 -am3699 -rf12345678 -x
TNSPay → Client	-tr03 -mc25530673 -am3699 -ms"Card Removed" -rc05 -rf12345678 -ed1301 -sys0 -x

5.7.10 Example: Mail Order Sale with Card Security Code

This example demonstrates the message exchange between TNSPay and a client application for a Mail Order (MOTO) Sale request.

The request provides only the mandatory tags identified in Table Section 5.7.1, and leaves it to TNSPay to fill all other authorisation data.

Table 5.18 Mail Order Sale with Card Security Code Example.

TNSPay Silent Workstation	
TNSPay ← Client	-rfNC123456 -tr09 -cd4539791001730106 -ed1301 -am2000 -sc 111 -x
TNSPay → Client	-tr09 -mc25530673 -am2000 -ac281569 -ms"Confirm Transaction" -qr5 -rc00 -rfNC123456 -tsl91 -tk200718812 -x
TNSPay ← Client	-ps0 -x
TNSPay → Client	-tr09 -mc25530673 -am2000 -ac281569 -ms"AUTH CODE:281569" -rc00 -ed1301 -rfNC123456 -pv0 -tk200718812 -sys0 -x
TNSPay Interactive Workstation	
TNSPay ← Client	-rfNC123456 -tr09 -cd4539791001730106 -ed1301 -am2000 -sc 111 -x
TNSPay → Client	-tr09 -mc25530673 -am2000 -ac281569 -ms"AUTH CODE:281569" -rc00 -ed1301 -rfNC123456 -pv0 -tk200718812 -

	sys0 -x

5.7.11 Example: E-Commerce Sale with Card Security Code & Address Verification

This example demonstrates the message exchange between TNSPay and a client application for an E-Commerce Sale request.

The request provides both the mandatory tags identified in Table Section 5.7.1, and the optional Address Verification tags(-hn, -zp).

Table 5.19 E-Commerce Sale with Card Security Code and Address Verification Example.

TNSPay Silent Workstation	
TNSPay ← Client	-rfEOS00172 -cd4444333322221111 -ed1301 -trB2 -am2000 -hn"12 Drury Street" -zp"ABC 123" -sc111 -x
TNSPay → Client	-trB2 -mc25530673 -am2000 -ac531384 -ms"Confirm Transaction" -qr5 -rc00 -rfEOS00172 -tsl91 -tk200718824 -x
TNSPay ← Client	-ps0 -x
TNSPay → Client	-trB2 -mc25530673 -am2000 -ac531384 -ms"AUTH CODE:531384" -ed1301 -rc00 -rfEOS00172 -pv0 -tk200718824 -sys0 -x
TNSPay Interactive Workstation	
TNSPay ← Client	-rfEOS00172 -cd4444333322221111 -ed1301 -trB2 -am2000 -hn"12 Drury Street" -zp"ABC 123" -sc111 -x
TNSPay → Client	-trB2 -mc25530673 -am2000 -ac531384 -ms"AUTH CODE:531384"-ed1301 -rc00 -rfEOS00172 -pv0 -tk200718824 -sys0 -x

5.7.12 Example: Mail-Order/Telephone-order (MOTO) (Keyed) transaction

Clients may use the in-built TPPC data-entry screens to capture card-data for MOTO transactions.
In this case, the CNP request should not contain any cardholder data.

Table 5.20 Mail-Order/Telephone-order (MOTO) (Keyed) transaction using TPPC for the data-capture

TNSPay Silent Workstation	
TNSPay ← Client	-rfEOS00172 -tr09 -am2000 -x
TNSPay → Client	-tsl91 -qr5 -ms"Confirm Transaction" -rc00 -pv0 -cnVISA -rfEOS00172 -am2000 -x
TNSPay ← Client	-ps0 -x
TNSPay → Client	-tr09 -mc6815145 -td03078912 -am2000 -ac099329 -ms"AUTH CODE:099329"-ed1301 -rc00 -rfEOS00172 -pv0 -tk21791 -sys0 -x
TNSPay Interactive Workstation	
TNSPay ← Client	-rfEOS00172 -tr09 -am2000 -x
TNSPay → Client	-tr09 -mc6815145 -td03078912 -am2000 -ac099329 -ms"AUTH CODE:099329" -rc00 -ed1301 -rfEOS00172 -pv0 -tk21791 -sys0 –x

5.7.13 Keyed Voice Authorisation (Referral) Sale Transaction

A voice Authorisation Transaction is considered by TPPC to be a Capture (Settlement) request as the authorisation has already been received by phone. For this reason, you must include the –pay tag, as well as certain other tags which are optional for a standard authorisation transaction. All of the tags shown below are required to perform a successful voice authorisation sale transaction.

Table 5.21 Keyed Referral (Voice Authorisation) Sale Transaction

TNSPay Silent Workstation	
TNSPay ← Client	-trz1 -rfIC12346 -cd4000000000000002 -ed1301 -mc6815145 -am1300 -cc826 -cu826 -ac12345 -pay –x
TNSPay → Client	-tsl91 -qr5 -ms"Confirm Transaction" -rc00 -pv1 -cnVISA -rfIC12346 -am1300 -x
TNSPay ← Client	-ps0 –x
TNSPay → Client	-trz1 -mc6815145 -td90008620 -am1300 -ac12345 -ms"AUTH CODE: 12345" -rc00 -ed1301 -rfIC12346 -pv1 -sys0 -x
TNSPay Interactive Workstation	
TNSPay ← Client	-trz1 -rfIC12346 -cd4000000000000002 -ed1301 -mc6815145 -am1300 -cc826 -cu826 -ac12345 -pay –x
TNSPay → Client	-trz1 -mc6815145 -td90008620 -am1300 -ac12345 -ms"AUTH CODE: 12345" -rc00 -ed1301 -rfIC12346 -pv1 –sys0 -x

5.7.14 Example: Keyed Voice Authorisation (Referral) Sale With Cash Back Transaction

A voice Authorisation Transaction is considered by TPPC to be a Capture (Settlement) request as the authorisation has already been received by phone. For this reason, you must include the –pay tag, as well as certain other tags which are optional for a standard authorisation transaction. All of the tags show below are required to perform a successful sale with cash back voice authorisation transaction.

Table 5.22 Keyed Referral (Voice Authorisation) Sale Transaction

TNSPay Silent Workstation	
TNSPay ← Client	-trz4 -cs500 -rfIC123456 -cd4000000000000002 -ed1301 -mc6815145 -am1300 -cc826 -cu826 -ac12345 -pay -x
TNSPay → Client	-tsl91 -qr5 -ms"Confirm Transaction" -rc00 -pv1 -cnVISA -rfIC127456 -am1300 -cs500 -x
TNSPay ← Client	-ps0 –x
TNSPay → Client	-trz4 -mc6815145 -td90008620 -am1300 -ac12345 -cs500 -ed1301 -ms"AUTH CODE: 12345" -rc00 -rfIC127456 -pv1 -sys0 -x
TNSPay Interactive Workstation	
TNSPay ← Client	-trz4 -cs500 -rfIC123456 -cd4000000000000002 -ed1301 -mc6815145 -am1300 -cc826 -cu826 -ac12345 -pay -x
TNSPay → Client	-trz4 -mc6815145 -td90008620 -am1300 -ac12345 -ed1301 -cs500 -ms"AUTH CODE: 12345" -rc00 -rfIC127456 -pv1 -sys2 -x

5.7.15 Example: Read Track Data from a Card

This example demonstrates how to read track 2 data from a card before processing a transaction. You may use this command to check card details before proceeding with a transaction. If TPPC is configured to accept a pre-inserted card, you may then follow it immediately with an authorisation request transaction, without any need for "double dipping". If TPPC is not configured to accept a pre-inserted card, then an additional Remove Card blocking message will be transmitted before the final response.

Note that TPPC uses a nominal amount of 1 currency unit (-ac100) to initiate the transaction on the terminal and this amount is included in both the status message and operator prompt. This amount should be ignored by the client and will not be include in the final response.

Table 5.23 Read Track2 Data Before Transaction Example.

TNSPay Silent Workstation	
TNSPay ← Client	-mg1 -x
TNSPay → Client	-tr10 -am100 -ms"Insert/swipe card" -rc00 -tsl10 -x
TNSPay → Client	-tr10 -am100 -ms"Checking Card" -rc00 -tsl20 -x
TNSPay → Client	-rc00 -cd5413330089000187=0712201066050169 -x
TNSPay Interactive Workstation	
TNSPay ← Client	-mg1 -x
TNSPay → Client	-rc00 -cd5413330089000187=0712201066050169 -x

5.7.16 Example: ICC Sale Fallback to Keyed: VSP-Enabled

This example demonstrates the message exchange between TNSPay and a client application for an ICC request, where the operator has chosen to key the transaction on a VSP-Enabled terminal.

The request provides only the mandatory tags identified in Table Section 5.7.1, and leaves it to TNSPay to fill all other authorisation data.

Table 5.24 ICC Sale to Keyed on a VSP Enabled Terminal Example.

TNSPay Silent Workstation	
TNSPay ← Client	-rfIC123456 -td03160017 -tr10 -am1500 -cc826 -cu826 -ds1"EMV Transaction" -ds2"ICC Request" -x
TNSPay → Client	-tr10 -am1500 -ms"Insert/swipe card" -rfIC123456 -tsl10 -x
TNSPay ← Client	-cmd00 -x
TNSPay → Client	-tr10 -am1500 -ms"Follow instructions on terminal" -rfIC123456 -tsl11 -x
TNSPay → Client	-tr10 -am3699 -ms"Cannot read card" -rc30 -rf12345678 -sys0 -x
Operator now punches in the card number and expiry date into the EMV terminal	
TNSPay → Client	-tr20 -mc25530673 -am1500 -ms"Authorising transaction" -rfIC123456 -tsl70 -x
TNSPay → Client	-tr20 -mc25530673 -am3699 -ms"Authorising transaction" -rf12345678 -tsl70 -x
TNSPay → Client	-tr20 -mc25530673 -am1500 -ac087814 -ms"Confirm Transaction" -qr5 -rc00 -rfIC123456 -tsl91 -pv1 -tk200723203 -x
TNSPay ← Client	-ps0 -x
TNSPay → Client	-tr20 -mc25530673 -am1500 -ac087814 -ms"AUTH CODE:087814" -rc00 -rfIC123456 -pv1-ed1301 -tk200723203 -sys0 -x

5.7.17 Example: Pre Authorisation

This example demonstrates the message exchange between TNSPay and a client application for a pre-authorisation ICC request.

Note: -sx1 flag must be set in a pre-authorisation request to prevent the transaction being presented for settlement.

The request provides only the mandatory tags identified in Table Section 5.7.1, and leaves it to TNSPay to fill all other authorisation data.

Table 5.25 Pre-Authorisation ICC Request Example.

TNSPay Silent Workstation	
TNSPay ← Client	-trY1 -am3699 -rf12345678 -sx1 -x
TNSPay → Client	-trY1 -am3699 -ms"Insert/swipe card" -rf12345678 -tsl10 -x
TNSPay ← Client	-cmd00 -x
TNSPay → Client	-trY1 -mc6815145 -am3699 -ms"Waiting for PIN entry" -qr2 -rf12345678 -tsl40 -x
TNSPay → Client	-trY1 -mc6815145 -am3699 -ms"PIN stage complete" -qr99 -rf12345678 -tsl60 -cvm410302 -x
TNSPay → Client	-trY1 -mc6815145 -am3699 -ms"Authorising transaction" -rf12345678 -tsl70 -cvm410302 -x
TNSPay → Client	-trY1 -mc6815145 -am3699 -ac005680 -ms"Confirm Transaction" -qr5 -rc00 -rf12345678 -tsl91 -pv0 -tk14741 -cvm410302 -x
TNSPay ← Client	-ps0 -x
TNSPay → Client	-trY1 -mc6815145 -am3699 -ac005680 -ms"Please remove card" -rc00 -rf12345678 -tsl80 -pv0 -tk14741 -cvm410302 -x

TNSPay → Client	-trY1 -mc6815145 -am3699 -ac005680 -ms"AUTH CODE:005680" -rc00 -rf12345678 -pv0 -tk14741-ed1301 -cvm410302 -sys0 -x

5.7.18 Example: ICC Sale with Card Acceptance and Surcharges

This example demonstrates the message exchange between TNSPay and a client application when Card Acceptance and Surcharges are enabled via TNSPay Payment Client Admin.

Table 5.26 ICC Sale with Card Acceptance and Surcharges.

TNSPay Silent Workstation	
TNSPay ← Client	-tr10 -rf12345678 -am3699 -x
TNSPay → Client	-tsl10 -ms"Insert/swipe/tap card" -rf12345678 -am3699 -x
TNSPay → Client	-tsl20 -ms"Checking Card" -rf12345678 -am3699 -x
TNSPay → Client	-tsl93 -qr6 -ms"Confirm cardtype" -ctctD -bin679999 -bln19 -cnMSTO -rf12345678 -am3699 -x
TNSPay ← Client	-prc00 -x
TNSPay → Client	-tsl94 -qr10 -ms"Apply Surcharge" -ctctD -bin679999 -bln19 -cnMSTO -rf12345678 -am3699 -cs0 -x
TNSPay → Client	-tr10 -rf12345678 -am5699 -sch2000 -prc3 -x
TNSPay → Client	-tsl40 -qr2 -ms"Waiting for PIN entry" -rf12345678 -am5699 -x
TNSPay → Client	-tsl60 -qr99 -ms"PIN stage complete" -rf12345678 -am5699 -x
TNSPay → Client	-tsl70 -ms"Authorising transaction" -rf12345678 -am5699 -x
TNSPay → Client	-tsl91 -qr5 -ms"Confirm Transaction" -rc00 -pv0 -rf12345678 -am5699 -x
TNSPay ← Client	-ps0 -x
TNSPay → Client	-tsl80 -ms"Please remove card" -rc00 -pv0 -rf12345678 -am5699 -x
TNSPay → Client	-tr10 -mc25530673 -td03160017 -am5699 -ac007956 -ms"AUTH CODE:007956" -rc00 -cnMSTO -ed1301 -rf12345678 -pv0 -tk17859 -cvm410302 -sys0 -x
TNSPay Interactive Workstation	
TNSPay ← Client	-tr10 -rf12345678 -am3699 -x
TNSPay → Client	-tsl93 -qr6 -ms"Confirm cardtype" -ctctD -bin679999 -bln19 -cnMSTO -rf12345678 -am3699 -x
TNSPay ← Client	-prc00 -x
TNSPay → Client	-tsl94 -qr10 -ms"Apply Surcharge" -ctctD -bin679999 -bln19 -rf12345678 -am3699 -cs0 -x
TNSPay ← Client	-tr10 -rf12345678 -am5699 -sch2000 -prc3 -x
TNSPay → Client	-tr10 -mc25530673 -td03160017 -am5699 -ac008013 -ms"AUTH CODE:008013" -rc00 -cnMSTO -rf12345678 -pv0 -tk17861 -cvm410302 -sys0 -ed1301 -x

5.7.19 Example: Transaction Tokenization

These examples demonstrate the message exchange between TNSPay and a client application for messages involving transaction tokenization.

5.7.19.1 Transaction-Token request

The tokenization process involves initially requesting a token for a sale-transaction. This is done by specifying the –toc mnemonic with a value of ‘10’:

Table 5.27 Request transaction token as part of a sale

TNSPay Silent Workstation	
TNSPay ← Client	-toc10 -rfIC123456 -tr10 -am1450 -cc826 -cu826 -x
TNSPay → Client	-tsl10 -ms"Insert/swipe card" -rfIC123456 -am1450 -x
TNSPay → Client	-tsl20 -ms"Checking Card" -rfIC123456 -am1450 -x
TNSPay → Client	-tsl40 -qr2 -ms"Waiting for PIN entry" -cnMCRD -rfIC123456 -am1450 -x
TNSPay → Client	-tsl60 -qr99 -ms"PIN stage complete" -cnMCRD -rfIC123456 -am1450 -x
TNSPay → Client	-tsl70 -ms"Authorising transaction" -cnMCRD -rfIC123456 -am1450 -x
TNSPay → Client	-tsl91 -qr5 -ms"Confirm Transaction" -rc00 -pv0 -cnMCRD -rfIC123456 -am1450 -x
TNSPay ← Client	-ps0 -x
TNSPay → Client	-tsl80 -ms"Please remove card" -rc00 -pv0 -cnMCRD -rfIC123456 -am1450 -x

TNSPay → Client	-tr10 -mc25530673 -td03160017 -am1450 -ac008821 -ed1301 -ms"AUTH CODE:008821" -rc00 -rfIC123456 -pv0 -tk22369 -cvm410302 -sys0 **-tov5413339899630012 -srcT0** -x
TNSPay Interactive Workstation	
TNSPay ← Client	-toc10 -rfIC123456 -tr10 -am1450 -cc826 -cu826 -x
TNSPay → Client	-tr10 -mc25530673 -td03160017 -am1450 -ac008827 -ed1301 -ms"AUTH CODE:008827" -rc00 -rfIC123456 -pv0 -tk22370 -cvm410302 -sys0 **-tov5413331022620012 -srcT0** -x

5.7.19.2 Transaction-Token use

A transaction token may be used later in a refund request by supplying the –toc mnemonic with a value of '11':

Table 5.28 Request transaction token as part of a sale

TNSPay Silent Workstation	
TNSPay ← Client	-tr61 -toc11 -am1450 -mc25530673 -rfIC123457 -tov5457216109943629 -x
TNSPay → Client	-tsl70 -ms"Authorising transaction" -cnMCDB -rfIC123457 -am1450 -x
TNSPay → Client	-tsl91 -qr5 -ms"Confirm Transaction" -rc00 -pv1 -cnMCDB -rfIC123457 -am1450 -x
TNSPay ← Client	-ps0 -x
TNSPay → Client	-tsl80 -ms"Please remove card" -rc00 -pv0 -cnMCRD -rfIC123456 -am1450 -x
TNSPay → Client	-tr61 -mc25530673 -td03160017 -am1450 -ac18930 -ms"AUTH CODE:18930" -rc00 -rfIC123457 -pv1 -tk22376 -sys0 -tov5457216109943629 -srcT0 -x
TNSPay Interactive Workstation	
TNSPay ← Client	-tr61 -toc11 -am1450 -mc25530673 -rfIC123457 -tov5457216109943629 -x
TNSPay → Client	-tr61 -mc25530673 -td03160017 -am1450 -ac21010 -ms"AUTH CODE:21010" -rc00 -rfIC123457 -pv1 -tk22375 -sys0 -tov5457216109943629 -srcT0 -x

5.7.20 Example: Card Tokenization

These examples demonstrate the message exchange between TNSPay and a client application for messages involving card tokenization.

5.7.20.1 Register A Card Token

In this example a request submitted to register a card for future use.

TNSPay Server	
TNSPay ← Client	-rfTNS000021 -mc25530673 -toc21 -cd4000000000000002 -ed1309 -x
TNSPay → Client	- tov4000002460520002 -srcT0 -rc00 -ms"Card saved successfully" –x

5.7.20.2 register A Card Token As part Of A Sale Transaction

In this example a **Card** Token is requested as part of a **Cardholder Not Present** Sale transaction.

TNSPay Server	
TNSPay ← Client	-rfTNS000020 -mc25530673 -td03160017 -tr09 -toc20 -cd4000000000000002 -ed1309 -am2000 -x
TNSPay → Client	-tr09 - tov4000002460520002 -cnVISA -tk33858 -td03160017 -st8292 -srcT0 -rfTNS000020 -rc00 -ms"AUTH CODE:547466" -mc25530673 -ma1 -cu826 -ci9 -cc826 -am2000 -ac547466 –x

5.7.20.3 Use a Card Token As part Of A Sale Transaction

In this example a **Card Token** is used in pace of card details to process a **Cardholder Not Present** Sale transaction.

TNSPay Server	
TNSPay ← Client	-rfTNS000022 -mc25530673 -toc22 -tov4000002460520002 -tr09 -am1450 –x
TNSPay → Client	-tr09 -tk33860 -td03160017 -st8293 -srcT0 -rfTNS000022 -rc00 -ms"AUTH CODE:117418" -mc25530673 -ma1 -cu826 -cnVISA -ci9 -cc826 -am1450 -ac117418 –x

5.7.20.4 Update A Card Token

In this example a request submitted to update the expiry date of an card which is already registered.

TNSPay Server	
TNSPay ← Client	-rfTNS000024 -mc25530673 -tov4000002460520002 -ed1512 -toc24 -x
TNSPay → Client	-srcT0 -rc00 -ms"Card updated successfully" -ed1512 -x

5.7.20.5 Update A Card Token As part Of A Sale Transaction

In this example a request submitted to update the expiry date of an card which is already registered as part of a **Cardholder Not Present** Sale transaction.

TNSPay Server	
TNSPay ← Client	-rfTNS000023 -mc25530673 -tr09 -am2450 -toc23 -tov4000002460520002 -ed1512 -x
TNSPay → Client	-tr09 -tk33863 -td03160017 -st8295 -srcT0 -rfTNS000023 -rc00 -ms"AUTH CODE:636421" -mc25530673 -ma1 -ed1512 -cu826 -cnVISA -ci9 -cc826 -am2450 -ac636421 -x

5.7.20.6 De-Register A Card Token

In this example a request submitted to de-register (delete) an existing **Card Token**.

TNSPay Server	
TNSPay ← Client	-mc25530673 - rfTNS000025 -tov4000001248880002 -toc25 –x
TNSPay → Client	-srcT0 -rc00 -ms"Card and token deleted successfully" –x

5.7.21 Pennies for charity transaction

These examples demonstrate the message exchange between TNSPay and a client application for which Pennies for charity is active.

TNSPay Silent Workstation	
TNSPay ← Client	-rfIC123457 -tr10 -am1430 -cc826 -cu826 -x
TNSPay → Client	-tsl10 -ms"Insert/swipe/tap card" -rfIC123457 -am1430 -x
TNSPay → Client	-tsl20 -ms"Checking Card" -rfIC123457 -am1430 -x
TNSPay → Client	-tsl30 -qr1 -ms"Request Cashback" -cnJCBC -rfIC123457 -am1430 -x
TNSPay ← Client	-cs0 -x
TNSPay → Client	-tsl60 -qr99 -ms"Cashback complete" -cnJCBC -rfIC123457 -am1430 –x
TNSPay → Client	-tsl25 -ms"PenniesDonation" -cnJCBC -rfIC123457 -am1430 -x
TNSPay → Client	-tsl40 -qr2 -ms"Waiting for PIN entry" -cnJCBC -rfIC123457 -am1529 -x
TNSPay → Client	-tsl60 -qr99 -ms"PIN stage complete" -cnJCBC -rfIC123457 -am1529 -x
TNSPay → Client	-tsl70 -ms"Authorising transaction" -cnJCBC -rfIC123457 -am1529 -x
TNSPay → Client	-tsl91 -qr5 -ms"Confirm Transaction" -rc00 -pv0 -cnJCBC -rfIC123457 -am1529 -x
TNSPay ← Client	-ps0 -x
TNSPay → Client	-tsl80 -ms"Please remove card" -rc00 -pv0 -cnJCBC -rfIC123457 -am1529 -x

TNSPay → Client	-tr10 -ac187881 -aidA0000000651010 -aip7C00 -am1529 -aqE3DF1C15EE05A7953030^ -aqc9FCCD8600D8C566F -atc0007 -aucFFC0 -av000800 -cc826 -cd3540829999421012 -cdr0 -ci4 -cid40 -chn"JCB/ICHIRO " -cnJCBC -ctt00 -cu826 -cvm410302 -ed4912 -et22 -iac0010000000F860ACF800F860ACF800 -iad0701010460000000 -ic9FCCD8600D8C566F!7C00!0007!B5A5265A!0880008000!00!0701010460000000!A0000000651010!0200!40!410302!FFC0!F800!E0B8C8 -ma1 -mc6815145 -ms"AUTH CODE:187881" -pem32 -psn00 -rc00 -rfIC123457 -sd5001 -st1792 -tcbE0B8C8 -td03078912 -tk2000001093 -tsiF800 -pv0 -ttpF296 -tvr0880008000 -upnB5A5265A -ver0200 -apl"JCB" -sys0 -cdr0 -don99 -dos12345 -dom12345678 -x
TNSPay Interactive Workstation	
TNSPay ← Client	-rfIC123457 -tr10 -am1430 -cc826 -cu826 -x
TNSPay → Client	-tr10 -ac187897 -aidA0000000651010 -aip7C00 -am1529 -aqA2BD0EC17F71A5A03030^ -aqc9FCCD8600D8C566F -atc0007 -aucFFC0 -av000800 -cc826 -cd3540829999421012 -cdr0 -ci4 -cid40 -chn"JCB/ICHIRO " -cnJCBC -ctt00 -cu826 -cvm410302 -ed4912 -et22 -iac0010000000F860ACF800F860ACF800 -iad0701010460000000 -ic9FCCD8600D8C566F!7C00!0007!279EE557!0880008000!00!0701010460000000!A0000000651010!0200!40!410302!FFC0!F800!E0B8C8 -ma1 -mc6815145 -ms"AUTH CODE:187897" -pem32 -psn00 -rc00 -rfIC123457 -sd5001 -st1793 -tcbE0B8C8 -td03078912 -tk2000001098 -tsiF800 -pv0 -ttpF296 -tvr0880008000 -upn279EE557 -ver0200 -apl"JCB" -sys0 -cdr0 -don99 -dos12345 -dom12345678 -x

5.7.22 Example: Query Status Message

These examples demonstrate the message exchange between TNSPay and a client application where the status of the transaction will be subsequently queried.

Note the inclusion of the Unique Idenitier (-uid) mnemonic and GUID value in the initial transaction request from the client. TPPC will include the Unique Identifier in every subsequent response. The Client however is not required to include the Unique Identifier in any additional messages.

5.7.22.1 Sale Transaction Including a Unique Identifier

This transaction includes a Unique Identifier so that its status may subsequently be queried by a client application.

Table 5.29 ICC Sale including a Unique Identifier

TNSPay Silent Workstation	
TNSPay ← Client	-ca000000 -rfIC123456 -td03160017 -tr10 -am100 -cc826 -cu826 -ds1"EMV Transaction" -ds2"ICC Request" -mc25530673 -x
TNSPay → Client	-tr10 -mc25530673 -am100 -ms"Insert/swipe card" -rfIC123456 -tsl10 -uid38a52be49352453eaf975c3b448652f0 -x
TNSPay → Client	-tr10 -am100 -mc25530673 -rfIC123456 -tsl20 -ms"Checking Card" -x
TNSPay → Client	-tr10 -am100 -mc25530673 -rfIC123456 -tsl40 -qr2 -ms"Waiting for PIN entry" -uid38a52be49352453eaf975c3b448652f0 -x
TNSPay → Client	-tr10 -am100 -cvm410302 -mc25530673 -rfIC123456 -tsl60 -qr99 -ms"PIN stage complete" -uid38a52be49352453eaf975c3b448652f0 -x
TNSPay → Client	-tr10 -am100 -cvm410302 -mc25530673 -rfIC123456 -tsl70 -ms"Authorising transaction" -uid38a52be49352453eaf975c3b448652f0-x
TNSPay → Client	-tr10 -rc00 -am100 -tk200718780 -ac295182 -cvm410302 -mc25530673 -ms"Confirm Transaction" -rfIC123456 -tsl91 -qr5 -x
TNSPay ← Client	-ps0 -x
TNSPay → Client	-tr10 -mc25530673 -am100 -ac295182 -ms"Please remove card" -rc00 -rfIC123456 -tsl80 -pv0 -tk200718780 -cvm410302 -uid38a52be49352453eaf975c3b448652f0 -x
TNSPay → Client	-tr10 -mc25530673 -am100 -ac295182 -ms"AUTH CODE:295182" -rc00 -rfIC123456 -pv0 -tk200718780 -cvm410302 -sys0 -x
TNSPay Interactive Workstation	
TNSPay ← Client	-ca000000 -rfIC123456 -td03160017 -tr10 -am100 -cc826 -cu826 -ds1"EMV Transaction" -ds2"ICC Request" -mc25530673 -uid38a52be49352453eaf975c3b448652f0 -x
TNSPay → Client	-tr10 -mc25530673 -am100 -ac295182 -ms"AUTH CODE:295182" -rc00 -rfIC123456 -pv0 -tk200718780 -cvm410302 -sys0 -uid38a52be49352453eaf975c3b448652f0 -x

5.7.22.2 Query Status Message

This example queries the status of the transaction shown in the previous example. The Unique Identifier used in the previous transaction is required to query the transaction status.

Table 5.30 Query Status message

TNSPay Silent Workstation	
TNSPay ← Client	–cmd1 -uid38a52be49352453eaf975c3b448652f0 –x
TNSPay → Client	-td03160017 -ma1 -cc826 -mc25530673 -rc00 -tr10 -am100 -cmd1 –cmr1 -cmtPENDING -ac770714 -ci5 -rfIC123456 -uid3F2504E04F8941D39A0C0305E82C3301 –tk200718780 -ms"AUTH CODE:295182" -st0112 -cu826 –x

Section 6 Proxy Client Configuration

6.1 Introduction

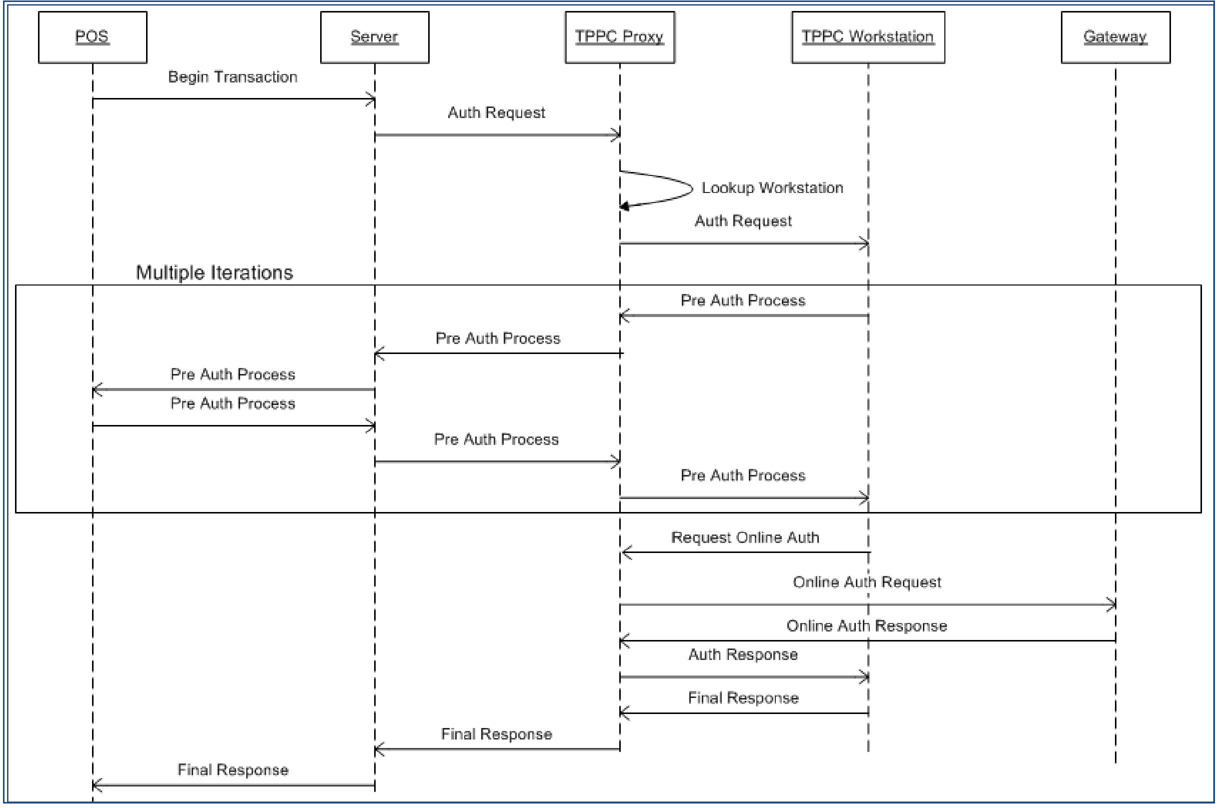
In Proxy Client mode, TPPC acts as a router and authorisation engine for a 3rd Party POS Server in an EMV CHP environment containing multiple TPPC Silent Workstation installations.

The POS Server will receive an initial transaction request from a 3rd Party POS client. The POS Server will forward the request to the TPPC Proxy Server, including in the request string the -dv mnemonic tag. It is recommended that the Device Serial Number for the EMV device associated with the TPPC Workstation instance be passed in this field, but any numeric value of between 1 and 10 digits may be used. This serves as a unique identifier to enable the Proxy Server to identify the IP Address and Port Number of a TPPC Workstation.

The Proxy Client forwards the request to the Workstation and acts as a proxy for all subsequent communications between the workstation and the POS Server.

The Proxy Server also acts as the authorisation server for the TPPC Workstation, in the same manner as the TPPC Server.

Fig. 6.1 Client Proxy Sequence (abbreviated).



6.2 Client Mapping

To enable the TPPC Proxy Server to identify an individual TPPC Workstation, a mapping file needs to be created by the implementer. This is an XML file in the format shown in Table 6.1.

Table 6.1 Client Mapping XML Schema.

```
<?xml version="1.0" encoding="utf-8" ?>
<clientmappings>
  <mapping clientid="1234">
    <targetip>127.0.0.1</targetip>
    <targetport>30501</targetport>
  </mapping>
  <mapping ...
  </mapping>
</clientmappings>
```

The name and location of the mapping file may be configured using the TPPC Admin application, and can be found under the Configuration tab.

Tables 6.2 & 6.3 describe each element in the client mapping XML schema provides an example file.

Table 6.2 Client Mapping Schema Definitions.

Client Mapping Schema	
Element	**Description**
clientmappings	The root node of the XML document.
mapping	Node containing a single TPPC Workstation mapping. The clientid attribute corresponds with the value to be submitted by the POS Server in the -dv tag.
targetip	The IP Address of the TPPC Workstation.
Targetport	The Port on which the TPPC Workstation will be listening.

Table 6.3 Sample Client Mapping File.

```
<?xml version="1.0" encoding="utf-8" ?>
<clientmappings>
  <mapping clientid="00244208">
    <targetip>192.168.2.1</targetip>
    <targetport>30501</targetport>
  </mapping>
  <mapping clientid="00244209">
    <targetip>192.168.2.2</targetip>
    <targetport>30501</targetport>
  </mapping>
  <mapping clientid="00244365">
    <targetip>192.168.2.25</targetip>
    <targetport>30503</targetport>
  </mapping>
  <mapping clientid="00241025">
    <targetip>192.168.2.101</targetip>
    <targetport>30502</targetport>
  </mapping>
</clientmappings>
```

Section 7 Generating Receipts

7.1 Overview

Receipt generation can be managed in two different ways. Either TNSPay Payment Client will automatically print a receipt as part of the transaction process flow, or it can pass back an XML document containing the receipt details.

This configuration can be set in TNSPay Admin. See the TNSPay Payment Client Administration Guide for details.

7.2 Printing a Receipt

When configured to print automatically, TNSPay Payment Client will print a receipt before it transmits the final transaction authorisation response. If no error is detected, TNSPay will assume that printing was successful and forward the final authorisation response. No print message is sent to the client if printing is successful.

If TNSPay Payment Client detects a printer error, it will send a print status query response message to the client. The client must respond to this message before the transaction can be completed, either to request a retry (-ps1) or to indicate that printing is complete (-ps0). Retries may continue until either TNSPay Payment Client detects no error, or a -ps0 is returned from the client.

Table 7.1 below shows a scenario where TNSPay continuously detects a print failure. The client twice requests a retry, and after the 3rd attempt, indicates that no further retries are required.

Table 7.1 Print Receipt Retries.

Print Error with Retry Response	
TNSPay → Client	`-ms"Printer Error. Retry?" -tsl91 -qr5 -x`
TNSPay ← Client	-ps1 -x
TNSPay → Client	-ms"Printer Error. Retry?" -tsl91 -qr5 -x
TNSPay ← Client	-ps1 -x
TNSPay → Client	-ms"Printer Error. Retry?" -tsl91 -qr5 -x
TNSPay ← Client	-ps0 -x

7.3 XML Receipt

7.3.1 Capturing Receipt Data

When configured to provide XML receipt data, the POS client controls the actual printing of the receipt. TNSPay Payment Client will automatically send a receipt message to the client before the final authorisation response is sent. This message contains both customer and merchant receipt copies.

TNSPay will wait for a print status response from the client before transmitting the final authorisation response. Table 7.2 shows a scenario where the operator sends back a retry request (-ps1) to the first receipt message received, and then send back a successful print status (-ps0) after the second attempt.

Table 7.2 XML Receipt Retries.

Print Error with Retry Response	
TNSPay → Client	-rfIC123456 -ms -rptc"<copy>* CUSTOMER RECEIPT *</copy><header2>PCI Test Department</header2><date>10/04/2013</date><time>17:42:51</time><reference>IC123456</reference><transactioncode>SALE</transactioncode><type>ICC</type><ctype>MASTERCARD</ctype><cardnumber>************1510</cardnumber><status>Approved 00</status><message>AUTH CODE:166841</message><amounttext>TOTAL SALE</amounttext><amount>GBP ?14.50</amount><accounttext>PLEASE DEBIT MY ACCOUNT</accounttext><verificationmethod>ICC PIN VERIFIED</verificationmethod><merchantno>**15145</merchantno><terminalno>****8912</terminalno><retain1>PLEASE RETAIN THIS COPY</retain1><retain2>THIS IS NOT A VAT RECEIPT</retain2><aid>A0000000041010</aid><psn>01</psn>" -rptm"<copy>** RETAILER COPY **</copy><header2>PCI Test Department</header2><date>10/04/2013</date><time>17:42:51</time><reference>IC123456</reference><transactioncode>SALE</transactioncode><type>ICC</type><ctype>MASTERCARD</ctype><cardnumber>************1510</cardnumber><status>Approved 00</status><message>AUTH CODE:166841</message><amounttext>TOTAL SALE</amounttext><amount>GBP ?14.50</amount><accounttext>PLEASE DEBIT MY ACCOUNT</accounttext><verificationmethod>ICC PIN VERIFIED</verificationmethod><merchantno>6815145</merchantno><terminalno>03078912</terminalno><aid>A0000000041010</aid><psn>01</psn>" -tsl71 -pv0 -x
TNSPay → Client	-tsl91 -qr5 -ms"Confirm Transaction" -rc00 -pv0 -cnMCRD -rfIC123456 -am1450 -x
TNSPay ← Client	-ps2 -x
TNSPay → Client	-rfIC123456 -ms -rptc"<duplicate>* DUPLICATE *</duplicate><copy>* CUSTOMER RECEIPT *</copy><header2>PCI Test Department</header2><date>10/04/2013</date><time>17:42:54</time><reference>IC123456</reference><transactioncode>SALE</transactioncode><type>ICC</type><ctype>MASTERCARD</ctype><cardnumber>************1510</cardnumber><status>Approved 00</status><message>AUTH CODE:166841</message><amounttext>TOTAL SALE</amounttext><amount>GBP ?14.50</amount><accounttext>PLEASE DEBIT MY ACCOUNT</accounttext><verificationmethod>ICC PIN VERIFIED</verificationmethod><merchantno>**15145</merchantno><terminalno>****8912</terminalno><retain1>PLEASE RETAIN THIS COPY</retain1><retain2>THIS IS NOT A VAT RECEIPT</retain2><aid>A0000000041010</aid><psn>01</psn>" -rptm"<duplicate>* DUPLICATE *</duplicate><copy>** RETAILER COPY **</copy><header2>PCI Test Department</header2><date>10/04/2013</date><time>17:42:54</time><reference>IC123456</reference><transactioncode>SALE</transactioncode><type>ICC</type><ctype>MASTERCARD</ctype><cardnumber>************1510</cardnumber><status>Approved 00</status><message>AUTH CODE:166841</message><amounttext>TOTAL SALE</amounttext><amount>GBP ?14.50</amount><accounttext>PLEASE DEBIT MY ACCOUNT</accounttext><verificationmethod>ICC PIN VERIFIED</verificationmethod><merchantno>6815145</merchantno><terminalno>03078912</terminalno><aid>A0000000041010</aid><psn>01</psn>" -tsl71 -pv0 -x
TNSPay → Client	-tsl91 -qr5 -ms"Confirm Transaction" -rc00 -pv0 -cnMCRD -rfIC123456 -am1450 -x
TNSPay ← Client	-ps0 -x

7.3.2 Receipt XML Tags

Table 7.3 outlines the XML tags included provided by TNSPay Payment Client.

Table 7.3 TNSPay XML Printing Tags.

Tag	Description
<duplicate>	DUPLICATE
<copy>	** RETAILER COPY ** OR "** CUSTOMER COPY **"
<header1>	Company Name
<header2>	Name
<header3>	Address Line 1
<Header4>	Address Line 2

<header5>	Address Line 3
<header6>	Address Line 4
<header7>	Address Line 5
<header8>	Address Line 6
<date>	Transaction Date
<time>	Transaction Time
<reference>	Transaction Reference
<type>	Transaction Type (ICC, Swipe, Contactless, Keyed)
<ctype>	Card Type (e.g. MCRD, VISA)
<cardnumber>	Card Number or Token
<aid>	Application Identifier
<psn>	Primary account number (pan sequence number)
<issueno>	Card Issue Number
<status>	Transaction Status (e.g. APPROVED 00)
<message>	Auth Code or reason for decline
<saleamounttext>	Sale Amount (GBP nn.nn)
<cashbacktext>	Cashback Amount (GBP nn.nn)
<surchargetext>	Surcharge Amount (GBP nn.nn)
<amounttext>	TOTAL SALE
<amount>	Total Transaction Amount (GBP nn.nn)
<accounttext>	e.g. "PLEASE DEBIT MY ACCOUNT"
<verificationmethod>	"CARDHOLDER NOT PRESENT", "ICC-PIN VERIFIED", "CARDHOLDER SIGNATURE VERIFIED", "NO CVM REQUIRED", "ICC-PIN + SIGNATURE VERIFIED"
<offlinespendingamount>	Offline Credit nn.nn
<signaturetext>	PLEASE SIGN BELOW
<retain>	PLEASE RETAIN THIS COPY
<retain2>	THIS IS NOT A VAT RECE
<merchantno>	Merhcant Number
<terminalno>	Terminal ID
<aqc>	Transaction cryptogram
<cvm>	Cardholder verification method list results
<tvr>	Terminal verification results
<iad>	Issuer application data
<tsi>	Transaction status information
<iac>	Issuer action codes
<charitytext>	Pennies for Charity text message
<ReceiptDonationText>	Pennies Donation Amount

7.3.3 Receipt Formatting Guidelines

If you choose to print your own receipts, they should fully conform to APACS rules (standard 70-1 H.2.2). See APACS documentation for further details.

The minimum height of any alphanumeric characters shall be 2.5mm and the number of characters per inch shall not exceed 15. The colour of the printed characters and backgrounds shall enable legibility in normal ambient light conditions. All carbon copies (where applicable) shall be legible.

Wording on receipts should not be modified or removed in any way as they follow requirements set in place by APACS, MasterCard and the acquirers.

Table 7.4 provides the suggested formatting for receipt fields. Where no formatting is suggested, the standard font left justified is recommended.

Table 7.4 Suggested Receipt Data Formatting.

Tag	Format	Details
header*n*	Centre	Up to 9 header lines may be included.
duplicate	Above Header, Centre	
type	Bold	Typical Values: Swipe ICC Keyed Keyed CNP
ctype	Bold	
cardnumber	Bold	
carddetails	Bold	
authmessage	Bold	
amounttext	Large	
amount	Large, Bold, Right justify	
footer*n*	Centre	Up to 9 footer lines may be included.

Section 8 TNSPay Payment Client PKI Protocol

8.1 Introduction

TNSPay Payment Client encrypts all TCP/IP communications between its components using a Public/Private Symmetric Key Exchange. This encryption process may also be used by integrators who want to provide additional security to communications between 3rd party applications and TNSPay Payment Client.

The encryption process may be accessed either through the use of the COMPlus component (See Section 9) or directly by developing your own component which complies with the TNSPay Payment Client PKI protocol.

Key sharing is *session* based, and a session may persist as long as both client and server application is running. Terminating either application will result in the termination of the PKI session.

8.2 PKI Protocol

8.2.1 Message Format

The protocol follows the standard request / response model. All messages have a common format, and consist of the following fields.

Table 8.1 TNSPay Payment Client PKI Message Protocol.

Name	Type	Description	HEX	Length (bytes)
Start of header (SOH)	byte	Indicates a packet start.	0x01	1
Session ID	int	The session ID of the current session (0 for PKI-Requests).		4
Status	int	Message status. 0 For success; positive integer for failure. If failure, a specific error message may be present in the data-segment.		4
Data Length	int	The length of the data-segment. This must be set to the exact number of bytes that are in the data segment ***after encryption.***		4
Start of Text (STX)	byte	Indicates the start of the data portion of the packet.	0x02	1
Data	byte[]	The binary data portion of the packet.		-
End Of Text (ETX)	byte	End of the data portion of the packet and end of packet.	0x03	1

8.2.2 Message Types

Two types of transaction may be sent using the PKI message protocol:

- Initialize Session Transaction
- Process Payment Transaction

8.2.3 Initialize Session Transaction

A PKI request is used to request a session id and shared symmetric key from the TNSPay Payment Client. During this process, an AES key is shared using a public/private key pair. A Session ID is also generated at this point and is returned with the shared AES key.

In order to set up the session with the server, clients are required to perform the following steps:

- Generate an RSA PKI key-pair. The RSA key length should be 256 bytes (2048 bit). You should make this length configurable to cater for changes to supported key lengths in the future.
- Send a new request with the following attributes:
    - Set the public-key created in (1) as the data segment.
    - Set the data-segment length (in the header) to the length (in bytes) of the public key
    - Set the Session ID to 0
- Read in the client response from the server.
- Store the Session ID in the response to be used in all subsequent requests in the session
- Decrypt the response data segment with the private key created in (1) using the RSA decryption algorithm.
- The result is a 48 byte-array. This is in fact a 32-byte session key followed by a 16-byte initialization vector, both of which should be stored for use in the session

Fig. 8.1 Initialize Session Sequence.

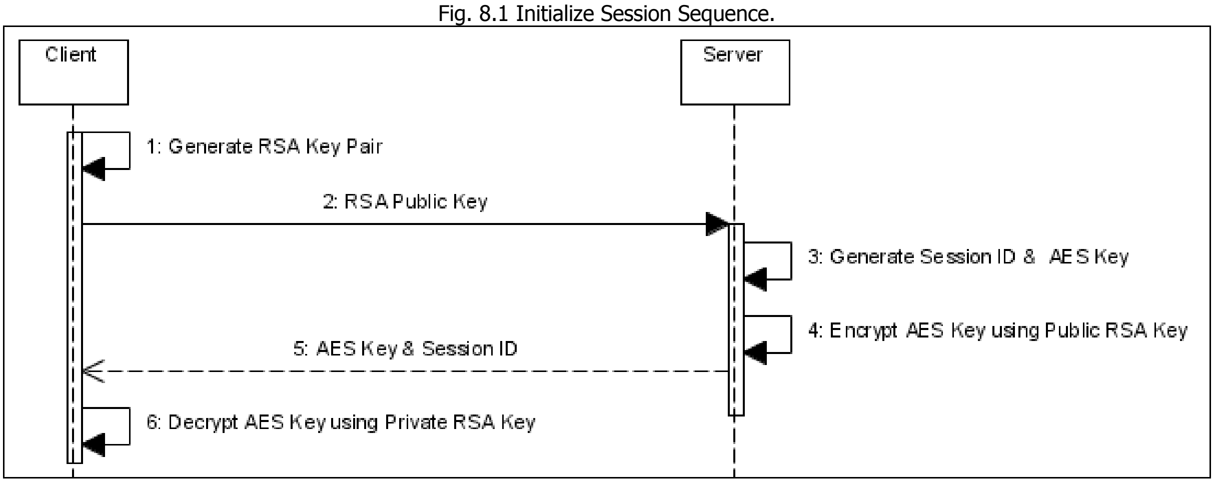

Client
Server
1: Generate RSA Key Pair
2: RSA Public Key
3: Generate Session ID & AES Key
4: Encrypt AES Key using Public RSA Key
5: AES Key & Session ID
6: Decrypt AES Key using Private RSA Key

Table 8.2 Initialize Session Request Message.

PKI Request	
Start Of Header	01
Session ID	00 00 00 00
Status	00 00 00 00
Length	14 01 00 00
Start Of Text	02
Public Key	06 02 00 00 00 A4 00 00 01 ...
End Of Text	03

Table 8.3 Initialize Session Response Message.

PKI Response	
Start Of Header	01
Session ID	01 00 00 00
Status	00 00 00 00
Length	00 01 00 00
Start Of Text	02
Symmetric Key (Encrypted)	43 48 78 3F DE C5 22 A0 F7 ...
End Of Text	03

8.2.4 Process Payment Transaction

A Process Payment Transaction is used to process authorisation and capture requests.

- Using the AES Key obtained in the Initialize Session Transaction, encrypt the request data, and add it to the data segment of a Message structure.
- Add the Session Key and length of the data segment to the message header.
- As with the Initialize Session Transaction response, the response data will be contained in the data segment, and a status code will be included in the header.

Fig. 8.2 Process Payment Transaction Sequence.

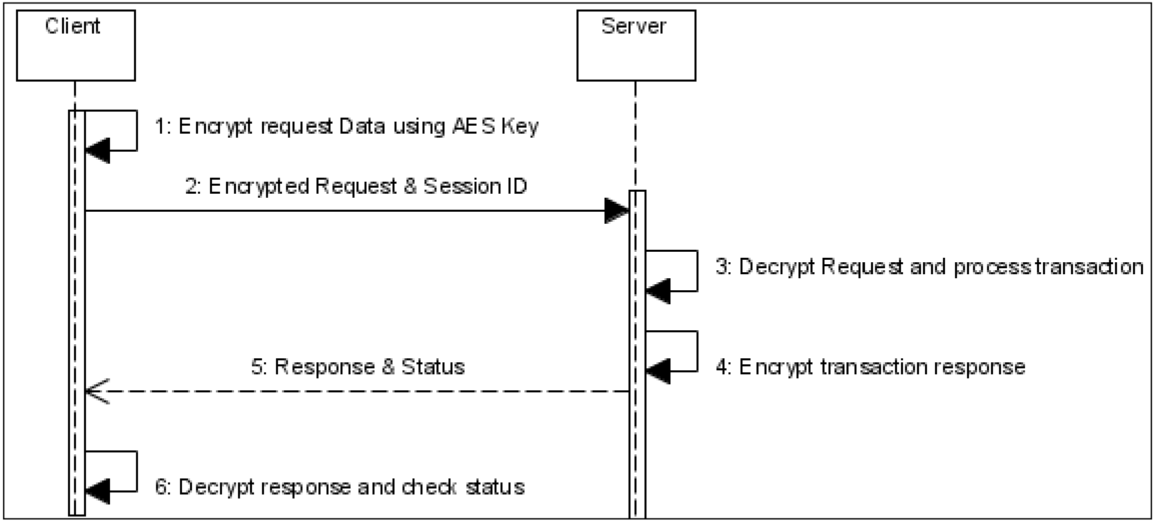

Client
Server
1: Encrypt request Data using AES Key
2: Encrypted Request & Session ID
3: Decrypt Request and process transaction
4: Encrypt transaction response
5: Response & Status
6: Decrypt response and check status

Table 8.4 Process Payment Request Message.

PKI Request	
Start Of Header	01
Session ID	01 00 00 00
Status	00 00 00 00
Length	40 00 00 00
Start Of Text	02
AES Encrypted Data	E6 CD DC 49 BB 98 EA C5 EB ...
End Of Text	03

Table 8.5 Process Payment Response Message.

PKI Response	
Start Of Header	01
Session ID	01 00 00 00
Status	00 00 00 00
Length	A0 01 00 00
Start Of Text	02
AES Encrypted Data	56 E3 64 A9 16 5D 73 30 70 ...
End Of Text	03

Section 9 COMPlus (TNSPay TCP Client Add-On)

9.1 What Is COMPlus?

COMPlus is an Add-On provided with TNSPay Payment Client designed to assist integrators to communicate with TNSPay Payment Client components. COMPlus provides a COM enabled .NET assembly (DLL) which provides basic TCP client functionality to the calling application. COMPlus also implements the TNSPay PKI protocol, which provides string encryption to all TCP packets transmitted between COMPlus and TNSPay Payment Client components.

As COMPlus was originally conceived to assist in migrating from a Flat File interface to a TCP/IP interface its interfaces use *string* rather than *binary* data.

9.2 Integrating with COMPlus

Integration with COMPlus requires either a .NET or COM compatible development language (e.g. C#.NET, VB.Net, VB6, Delphi, Python, VB Script, JScript).

9.2.1 Configuration

COMPlus requires the tnspayments.interop.config file to be located in the same directory as the executing application. Unlike standard TNSPay configuration files, this file may be edited directly in a text editor.

The following configuration options are available:

Table 9.1 COMPlus Configuration Settings.

Key	Value	Description
serverip	*n.n.n.n* e.g. 192.168.25.15	The IP address of the TNSPay Payment Client instance.
serverport	Integer	The port number on which the TNSPay Payment Client instance will be listening.
Aeskeylength	Integer	The length of the AES Key used for symmetric encryption. Do not change this value unless requested to do so by TNS.
aesivlength	Integer	The length of the AES Initialization Vector used for symmetric encryption. Do not change this value unless requested to do so by TNS.
rsakeylength	Integer	The length of the RSA Key used for asymmetric encryption. Do not change this value unless requested to do so by TNS.

9.2.2 Methods

Table 9.2 COMPlus Public Methods.

Name	Arguments	Return Type	Description
ProcessPayment	data : string	string	This method initiates a transaction with TNSPay. If a current shared key is not available this method will request a key from TNSPay.
Write	data : string	void	Sends a block of data to TNSPay.
GetResponse		string	Reads the COMPlus response buffer and returns its contents.
Close		void	Calls the Dispose method of COMPlus to clean up.
Dispose		void	Closes the TCP connection and frees any resources being used by COMPlus.

9.2.3 COM Interface

To integrate with TNSPay Payment Client using the COMPlus COM interface, you must instantiate a late bound reference to COMPlus through its IDispatch interface.

Table 9.3 COMPlus Identifiers.

Identifier	Value
CLSID	9A87598D-0DDA-40e7-944C-B6618F46A00B
PROGID	TnsPayments.Interop

The following examples example demonstrates how to create a COMPlus reference.

Table 9.4 Initialize COMPlus.

```
‘Visual Basic

Dim objComPlus as Object
Set COMPlus = CreateObject("TnsPayments.Interop")

//Javascript

var COMPlus = new ActiveXObject("TnsPayments.Interop");
```

To initiate a session with TNSPay Payment Client and COMPlus, call the ProcessPayment() method. This method takes one string argument, which should be a flexible format transaction request.

Table 9.5 Initialize a transaction.

```
‘Visual Basic

objComPlus.ProcessPayment (“-tr10 -rfSW123456 -am900 -x”)

//Javascript

COMPlus.ProcessPayment(“-tr10 -rfSW123456 -am900 -x”);
```

COMPlus does not provide an event for reading responses. To read a response you must call the COMPlus.GetResponse() method. This method takes no arguments and returns a flexible format string.

Table 9.6 Read the Response Buffer.

```
‘Visual Basic

Dim strResponse as string
strResponse = objComPlus.GetResponse()

//Javascript

var strResponse = objComPlus.GetResponse();
```

Finally to transmit additional data to the TNSPay Payment Client, use the Write() method. This method takes one string argument, which should be a flexible format transaction request.

Table 9.7 Send a Cashback Response.

```
‘Visual Basic

objComPlus.Write(“-cs5000 -x”)

//Javascript

objComPlus.Write(“-cs5000 -x”);
```

9.2.4 COMPlus .NET Interface

The COMPlus .NET Interface may be accessed from a .NET capable application by accessing the public methods of the Tns.Payments.Interop assembly. Using Microsoft Visual Studio 2008, simply add a project reference to this assembly, and you can call the COMPlus methods directly.

Appendix A - Flexible Format Mnemonics Specification Update

New Mnemonic Fields

The following new mnemonic fields, required for integration with TNSPay, are not yet documented in the TNS Flexible Format Mnemonics Specification.

Pennies Donation amount(don)
When a donation is made to the pennies charity (accept is confirmed on the pin), the amound for donation is stored in the don field.

Pennies Merchant Number (pom)
The identifier pennies use for distinguishing merchants. Note this is different from tppc merchant numbers.

Pennies Store Number (pos)
The identifier pennies use for distinguishing stores. Note this is different from tppc merchant stores.

Merchant Receipt XML (rptm)
The Merchant Receipt mnemonic contains the receipt data required to print a merchant copy of a receipt. This mnemonic is only used when TNSPay Payment Client is configured in Silent Workstation Mode with XML Receipt Generation enabled.

Customer Receipt XML (rptc)
The Customer Receipt mnemonic contains the receipt data required to print a customer copy of a receipt. This mnemonic is only used when TNSPay Payment Client is configured in Silent Workstation Mode with XML Receipt Generation enabled.

Print Verification (pv)
The Print-Verification mnemonic defines how the transaction should be verified with-respect to signature:

Table A1 Print Verification Values.

Value	Description
0	Signature not required.
1	Signature required.
2	Signature required AND PIN verified.

This mnemonic aids the 3rd-party client in knowing whether the request the operator to ask the customer to sign the receipt, as well as to know when the print a signature-strip on the receipt (XML-receipt format).

Process Status (-ps)

The Process Status message is used to determine the final process status between the POS Client and TNSPay Payment Client in Silent Workstation mode prior to finalising the transaction.

Table A2 Process Status Values.

Value	Description
0	Commit.
1	Void.
2	Reprint Receipts.
3	Reprint Merchant Receipt Only.
4	Reprint Customer Receipt Only.

Command Type (-cmd)

The Command Type message is used to send various commands to the TNSPay Payment Client. This mnemonic is only used when TNSPay Payment Client is configured in Silent Workstation Mode

Table A8 Command Type Values.

Value	Description
00	Key VSP card details (Used when waiting for card-entry: *-ts l10*).

System State (-sys)

The System State message is included in the final response from TPPC for every transaction and indicates the health of the online system. The value contained in this tag is a bitmask as more than one failure state may be detected by TPPC.

Table A9 System State Values.

Value	Description
0	OK.
1	Authorizer Offline.
2	Capture Failure.
4	Contactless Module Disabled

It is recommended that integrators check the system state after each transaction to determine if there are any issues that need to be highlighted to the operator.

For instance, a value of 1 indicates that the system is unable to communicate with the TPRG which will result in a much higher level of declined transactions.

A value of 2 indicates that capture (payment) records have been rejected by the TPRG. This can indicate a serious problem with the TPPC configuration, and needs to be addressed urgently. It is recommended that this value be "RED FLAGGED" immediately to the operator, and client operator instruction manuals/training should highlight the need for this system state to be investigated urgently.

A value of 4 indicates that Contacless Transactions are enabled in TPPC but it was not possible to enable the Contactless reader on the EMV Device.

POS Response Code (-prc)

The POS Response Code is used to respond to surcharge and card acceptance requests from TNSPay Payment Client to the POS client.

Response values of 0 and 1 allow the user to accept or reject the card type returned by TNSPay if card acceptance is configured in TNSPay Payment Client Admin.

Response values 2 to 4 are used to indicate whether a surcharge has been applied to an amount or not during the surcharge stage of a transaction only. Please see Section 5.7.15 for an example of where these values are used.

Table A5 POS Response Code Values.

Value	Description
0	Confirm Card Type.
1	Decline Card Type.
2	No Surcharge Applied.
3	Surcharge Applied or Declined.
4	Decline Surcharge.

Surcharge Amount (-sch)

The Surcharge Amount field is used to specify an amount which has been added to the original transaction amount during the surcharge stage if enabled in TNSPay Payment Client Admin.

If an amount is to be added it must be accompanied by the transaction reference and the relevant –prc value to indicate its status. The original amount must also be amended to reflect the addition of the surcharge before submission. Please see Section 5.7.15 for an example of how this mnemonic is used.

Tokenization command (-toc)

The Tokenization command mnemonic is used request or use a token

Table A6 Tokenization Command Values.

Value	Description
10	Request Transaction Token
11	Use transaction Token

New Mnemonic Values

The following table contains updates to the values accepted in certain mnemonic fields.

Table A7 New Mnemonic Values.

Field	Value	Description
-tsl	11	Waiting for VSP Keyed-entry on Terminal.
-tsl	91	TNSPay Payment Client Confirm Transaction.
-tsl	92	Voice Authorisation Required.
-tsl	93	Confirm Cardtype
-tsl	94	Apply Surcharge
-tsl	25	Inviting cardholder to donate
-qr	5	TNSPay Payment Client Confirm Transaction Status.
-qr	9	Voice Authorisation (Referral) Auth Code Required.
-qr	10	Apply Surcharge