XML Application Programming Interface A(PI) Specifications for POSLynx with TransNet

API Guide

Version 4.0

Table of Contents

Chapter 1: Overview

Introduction

This document describes the Extensible Markup Language (XML) used to execute transactions using the POSLynx XML Application Programming Interface (API).

The Precidia XML API is intended to be an efficient, flexible, and relatively straight forward method of processing various transaction types for a variety of acquirers or hosts. It is designed to be both generic and host-agnostic, although some elements may be ignored for certain hosts, but deemed mandatory for others.

The API is not specific to any single POSLynx device in the product family. The API is accepted by all POSLynx models such as the POSLynxMINI or the POSLynxDUO.

NOTE: The XML examples shown in this document are for illustration purposes only and contain fictional card numbers, merchant details, and other data not intended to be used in real-world scenarios.

Comments, requests, and suggestions for both this document and the API are welcome, and can be sent to salesgroup@precidia.com.

The Precidia API is continually evolving and amended when new hosts and/or transaction types are introduced. Therefore, please access Precidia's Partner Support portal at http://help.precidia.com/ to obtain the latest and most up-to-date version of this document.

Background Information

The most common method of interfacing with the POSLynx using the XML API described in this document is communicating using the Precidia TransNetPOS-CG helper application. This application (available on both Windows and Linux platforms) allows the POS application to communicate locally with the POSLynx device. This method provides certain advantages such as a facility to allow the TNP-CG to deal with credit card entry (manual or swiped) via its own GUI. Please refer to Figure 1: The TransNetPOS-CG (TNP-CG) application, for a view of the GUI.

NOTE: Throughout this document, all element tags highlighted in red are only applicable to TNP-CG.

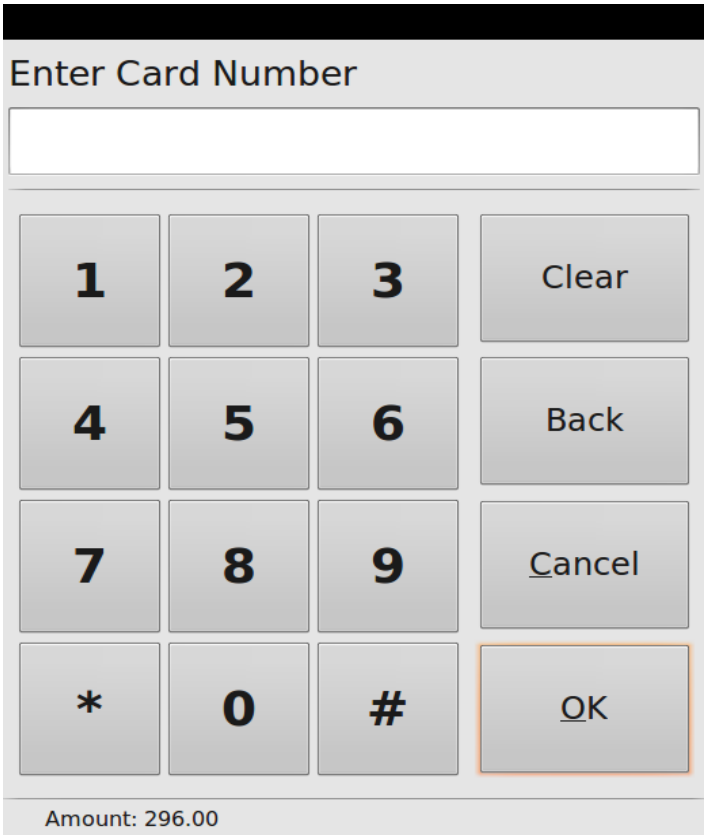


Figure 1: The TransNetPOS-CG (TNP-CG) application GUI

For further information on the TransNetPOS-CG and other options available for connecting and sending requests to the POSLynx, please refer to the document entitled "Using the TransNetPOS-CG to Access the XML API for POSLynx220™ with TransNet™", accessible from the http://help.precidia.com/ Partner Support website.

Chapter 2: Communication

Communication Interface

There are several methods available for sending XML requests to the POSLynx. These choices offer the developer a great deal of flexibility in using the API and integrating it with the POS application.

TransNetPOS-CG

The TransNetPOS-CG (TNP-CG) is a 'helper' application which can be installed on the same system as the Point of Sale (POS) application. The POS communicates with the TNP-CG and the TNP-CG in-turn communicates (securely) with the POSLynx. The TNP-CG provides additional functionality such as prompting for card data and an end-of-day GUI. Please see "Using the TransNetPOS-CG to Access the XML API for POSLynx220™ with TransNet™" on http://help.precidia.com for further details. An ActiveX/OCX Control providing similar functionality is also available.

NOTE: Throughout this document, all element tags highlighted in red are only applicable to TNP-CG.

Communication Options

The methods of communication that are available are outlined in the following sections.

Local TCP/IP Socket

This option involves placing a small helper application (TNP-CG) provided by Precidia Technologies on the POS system, which will handle the sending of requests via SSL sockets to the POSLynx device. The POS system can thus communicate with this application, sending and receiving XML requests/responses via this socket.

File-Drop

The file-drop method involves installing a small application designed by Precidia Technologies on the POS system, which will monitor a local directory. Files containing XML requests are written to this directory by the POS. Each request is written to a separate file. The resulting responses are subsequently written as files in the same directory. The local application will handle sending the requests via SSL sockets to the POSLynx. This method is perhaps the easiest to use, however is potentially not as secure as the other communication options.

Embedded ActiveX Control

An ActiveX OCX Control can be used to send XML requests to the POSLynx. This Windows-based POS application can embed the control and make 'function calls' to the local ActiveX control. The ActiveX control will then open a secure SSL connection to the POSLynx to

process transactions. Once set-up, this embedded control is a simple method of processing requests and receiving responses.

Direct TCP/IP Socket

The POSLynx acts as a server, with the client opening a direct connection on a configurable port and passing the XML over the socket. The XML response is then returned via the same socket. An SSL connection is recommended. In most situations, it is also recommended that the TNP-CG application is used to write to a local socket (rather than directly communicating with the POSLynx).

Client MAC Address

The POSLynx expects to receive a Client MAC address within the XML request. In most scenarios, the TNP-CG will determine this value and include it in the request it sends to the POSLynx. However, when using a direct connection to the POSLynx, the Client MAC address must be explicitly set within the XML request using the following XML element:

```
<ClientMAC>001122334455</ClientMAC>
```

Chapter 3: Host Processor

Host Processor Handling

The POSLynx device's configuration contains details of the host processor setup, including both Merchant ID and Terminal ID. Therefore, there is no requirement for these details to be included with the XML transactions.

As previously mentioned, the XML elements forming a request are intended to be host-agnostic, allowing the POSLynx to be reconfigured with a new host with no knock-on effect to the POS system itself.

Chapter 4: Errors

Error Handling

The <*ResultText*> field, indicating an error or declined transaction, is included with most failed response messages. The <*ResultText*> field will provide more details of the error or a reason for the declined/failed transaction.

The content of the <*ResultText*> field is usually a text-string returned by the host. Therefore, different hosts may return different text strings for a particular declined/failed transaction scenario. Conversely, error messages originating from the POSLynx, will be consistent. For example, when the host does not respond, the text-string 'HOST TIMEOUT' is returned.

Errors typically occur under several possible scenarios described below:

1. Failure of the POSLynx to respond in time (usually leads to ErrorCode 0130 – PIN Pad timeout).
2. Device communication failure.
3. Host timeout.
4. Incorrect input.

Error Codes

The following tables define the error codes returned from the POSLynx.

A successful transaction contains a <*Result*> value of APPROVED (or AP) and an <*ErrorCode*> value of "0000".

Should the transaction be declined by the host processor, the <*Result*> value will be DECLINED and the <*ErrorCode*> will be of value 0001.

Transaction Error Codes

Many of these errors will NOT occur when using the XML interface, but are included for completeness.

Error	Definition
0000	Transaction Approved (i.e. no error)
0001	Transaction Declined – See Host response in *<ResultText>*
0002	Invalid Amount
0003	Record Not Found
0004	Incorrect Amount
0008	Account Number Not Sent
0009	Unsupported Card
0010	Invoice Number Missing
0011	Expired Card
0014	CVV value Missing
0016	Invalid PIN
0017	PIN Block Missing
0019	PAN Mismatch
0055	Incorrect PIN
0058	Invalid Transaction
0077	Rejected Batch
0078	Transaction Not found (e.g. Void command no matching Record Number)
0079	Declined due to bad CVV value
0080	Bad Batch Number
0085	Batch Not Found
0089	Bad Terminal ID
0090	Close Batch before processing transactions – previous batch close likely failed
0091	Batch Out of Balance
0092	Transaction Error – Default if other errors do not apply.
0093	Transaction Not Complete
0094	Transaction Not Confirmed
0095	Transaction Re-Macing required
0096	Transaction Not Executed
0097	Transaction Reversed
0098	Duplicate Transaction
0099	Unknown Transaction
0100	User aborted Transaction
0101	Network Down
0102	Host Timeout – no response received from host processor
0103	Incomplete Transaction

0104	No Route to Host Defined
0105	Invalid Record Number
0106	Service Not Supported
0107	Bad Card Number (failed Module10 Check)
0108	Card is Expired
0109	Invalid Expiry Date
0110	Invalid Time on POSLynx
0111	Invalid Track Data
0112	Transaction Reversed - Try again later
0113	Call Help Desk (of Card Processor)
0114	Amount not Matched
0115	Invalid Adjust Amount
0116	Repeated Transaction (not always flagged depends on options set)
0117	Resend the Result
0118	Loyalty not Allowed
0119	Batch Version Invalid
0120	Invalid Batch Number
0121	Failure to Change Batch Number
0122	Failure to Open Batch
0123	Failure to Clear/Remove Batch
0124	Previous Batch Settled
0125	Failure to Update Batch
0126	Batch is Locked
0127	Host Initiated Settlement Only
0128	Batch Has Negative Balance
0129	Batch has Zero Items
0130	PIN Pad Error
0131	No MSR
0132	No MCR
0133	Terminal Timeout
0134	Invalid Format
0135	Transaction Not Supported
0136	Request Ignored
0137	No Record to be Reversed
0138	Reversal Cleared
0139	Host Not Ready
0140	Datawire Host Fatal Error
0141	Backend Host Failed
0142	Failed Connecting to Host
0143	Hold Cards and Call or Pick up Cards
0144	Do Not Honour
0145	Re-enter Last Name
0146	Re-enter SIN

0147	Re-enter IDs
0148	Record Active
0149	Record Locked
0150	Record Not Active
0151	Record is Void
0152	Not Enough Loyalty Points
0153	Batch Group/Record Status Error
0154	Batch Group/Record Id Already Exists
0155	End of Sequence or Sequential Operation Error
0156	Out of Memory
0157	Group/Record Id Not Found
0158	Batch Group Not Open
0159	Retrieved Request Invalid
0160	Batch is Empty Cannot Release it
0161	POSLynx Reset During Operation
0162	End of Batch Reached During Browsing
0163	No More Records to Browse
0164	Invalid Server Id
0165	Batch is Partially Settled
0166	EMV Table is Empty

Table 1

Invalid Input Error Codes (XML Parsing Errors)

Error	**Definition**
1000	Error Parsing XML – Catch-all error if serious error occurs
1001	No <*PLRequest*> Tag found
1002	Missing or Invalid/Unsupported <*Command*> specified
1003	Missing or Invalid <*Amount*> specified
1004	Missing or Invalid <*Input*> specified
1005	Missing or Invalid <*Track2*> specified
1006	Missing or Invalid <*CardNumber*> specified
1007	Missing or Invalid <*ExpiryDate*> specified
1008	Missing or Invalid <*RecNum*> specified
1009	Missing or Invalid <*ClientMAC*> specified
1010	Command Not Valid in Current Configuration or requires <*BatchIndex*> Element
1011	Missing or Invalid <*BatchNum*> specified
1012	No Pinpad Configured for Debit command
1013	Invalid/Missing Account Number

Table 2

Other Error/Response Codes

Many of these errors occur only when using the TransNetPOS-CG helper application to communicate with the POSLynx.

Error	Definition
2001	Unable to Connect to POSLynx
2002	Error Sending data to POSLynx
2003	Error Reading data from POSLynx
2004	Timeout Reading data from POSLynx
2010	Transaction Canceled by User
2011	User Input Timed Out
2020	Timeout Communicating with PIN Pad
2021	Error communicating with PIN Pad
2022	Transaction Canceled at PIN Pad
2023	Invalid PIN Pad settings specified
3001	Threaded Transaction started. Not an error condition, but interim success. Final result will be sent via event. (This only applies to TransNetPOS-CG ActiveX control)

Table 3

Chapter 5: Card Entry

Card Entry Options

The POS system typically allows the user to swipe a card or to manually enter a card number. However, the magnetic stripe reader could be attached to either the POS system or to the POSLynx. This *<Input>* element allows for such situations.

For example, when card information is entered manually, the XML command is:

```
<Input>MANUAL</Input>
```

For commands that require a card number, the following table outlines options that are available.

Option <Input>	Explanation	Additional XML Required
MANUAL	The card number and expiry date were manually entered on the POS system	*<CardNumber><ExpiryDate>*
SWIPED	The card was swiped on a POS MSR, the track1 and/or track2 data is sent as part of the request. Currently Track2 data has priority as it will be processed for all hosts. If both Track1 and Track2 data are present the name will be taken from the Track1 data, but the Track2 data will be used for card information.	*<Track1>* and/or *<Track2>*
EXTERNAL	The POS is not sending any card information and either the POSLynx or a helper application must perform the swipe. TransNetPOS-CG or TransNetPOS-CGAX can communicate with a PIN Pad connected to the POS.	None
TOKEN	A token is received in response to a transaction request, following which it can be used, in place of Card Number or Track2 data, in subsequent transactions.	*<ExpiryDate><Token>*
NONE (no *<Input>* parameter)	Only available if using TransNetPOS-CG application.	None

	This will cause TNP-CG to prompt for the card data, using its own GUI. Allows user to manually enter or swipe card data. Used when the *<Input>*, *<Track2>,* and*<CardNumber>* elements are not specified. See the TNP-CG document for further details	

Table 4

Chapter 6: Receipts

Generating Receipts

Should the POSLynx be configured with a printer specified on the ECR, the XML response will contain receipt information. The receipt is formatted by the POSLynx, depending on the type of transaction and the host processor configuration. Often, the host processor has specific requirements for the receipt generated.

There will be two receipts generated, one for the merchant and one for the customer, the difference usually being the signature line, which appears on the merchant copy only(although this is host-dependent). In the POSLynx receipt settings, rules can be established to govern when the signature line is added to a receipt, based on the dollar amount of the transaction.

Gift card receipts will display the remaining balance on the card used.

NOTE: The time of day that is listed on the receipt is set by the POSLynx. By default the POSLynx is set to EST (Eastern Standard Time), but can bechanged to any time zone, with daylight savings turned on or off.

The receipts will have the following format:

```
<Receipt>
    <Receipt1>first line to print</Receipt1>
    <Receipt2>second line to print</Receipt2>
...
    <Receiptn>nth line to print</Receiptn>
</Receipt>
<ReceiptCustomer>
    <Receipt1>first line to print</Receipt1>
    <Receipt2>second line to print</Receipt2>
...
    <Receiptn>nth line to print</Receiptn>
</ReceiptCustomer>
```

For example:

```
<Receipt>
   <Receipt1>TERM # 001</Receipt1>
```

```
   <Receipt2>TYPE PURCHASE</Receipt2>
   <Receipt3>ACCOUNT TYPE MC</Receipt3>
   <Receipt4>CARD NUMBER ************5454</Receipt4>
   <Receipt5>DATE/TIME 11/30/09 16:06:58</Receipt5>
   <Receipt6>REC # 10076</Receipt6>
   <Receipt7>REFERENCE # 00000000 S</Receipt7>
   <Receipt8>TRACE # A0001943</Receipt8>
   <Receipt9>AUTHOR. # 194372</Receipt9>
   <Receipt10>AMOUNT $1.00</Receipt10>
   <Receipt11>--------------</Receipt11>
   <Receipt12>TOTAL $1.00</Receipt12>
   <Receipt13>--------------</Receipt13>
   <Receipt14>APPROVED - THANK YOU</Receipt14>
   <Receipt15>IMPORTANT -- retain this copy for your records</Receipt15>
   <Receipt16 />
   <Receipt17 />
   <Receipt18>X_____</Receipt18>
   <Receipt19>CARDHOLDER WILL PAY CARD ISSUER ABOVE</Receipt19>
   <Receipt20>AMOUNT PURSUANT TO CARDHOLDER AGREEMENT</Receipt20>
</Receipt>

<ReceiptCustomer>
   <Receipt1>TERM # 001</Receipt1>
   <Receipt2>TYPE PURCHASE</Receipt2>
   <Receipt3>ACCOUNT TYPE MC</Receipt3>
   <Receipt4>CARD NUMBER ************5454</Receipt4>
   <Receipt5>DATE/TIME 11/30/09 16:06:58</Receipt5>
   <Receipt6>REC # 10076</Receipt6>
   <Receipt7>REFERENCE # 00000000 S</Receipt7>
   <Receipt8>TRACE # A0001943</Receipt8>
   <Receipt9>AUTHOR. # 194372</Receipt9>
   <Receipt10>AMOUNT $1.00</Receipt10>
   <Receipt11>--------------</Receipt11>
   <Receipt12>TOTAL $1.00</Receipt12>
   <Receipt13>--------------</Receipt13>
   <Receipt14>APPROVED - THANK YOU</Receipt14>
   <Receipt15>IMPORTANT -- retain this copy for your records</Receipt15>
   <Receipt16 />
</ReceiptCustomer>
```

NOTE: Since the XML is the same format for all transactions, the examples in the following sections will not contain either the *<Receipt>* or *<ReceiptCustomer>* elements.

Chapter 7: Transaction Reports

Detailed Transaction Reporting

In most cases, only the transaction total amount is sent to the host for approval. However, should the option be required, extra transaction detail (usually a listing of the items sold) can be included with the transaction request. The details sent can be reported using Precidia's MerchantVu web application.

Elements for *TranDetail*

The following table lists elements required for reporting transaction details.

Element Tag	Required	Type	Size	Notes
TranDetail	N	TAG	-	Top level tag includes the other tags shown below.
Item	Y	TAG	-	Text description of Item, if tax must be specifically labelled as "Tax". Contains sub tags representing an item on the receipt.
Quantity	Y	Int	1-5	Number of this items sold to this customer e.g. 2 Coffee and 1 Muffin sold.
Department	N	String	1-20	Department of item sold. e.g. Grocery, or Produce or Housewares. Can be an integer code.
PluNumber	N	String	1-20	Code or name for this specific item. e.g. Large Coffee or Muffin or Brand X 2 slice toaster All items of one PLU have the same price. For Tax, this will be "Tax".
SerialNo	N	String	1-20	Serial Number of the item. Optional as tracking of specific items sold may not be desired, but allows tracking of a single specific item if required
Price	Y	Float	3-7	Cost of this line item in transaction. Formatted with 2 decimal places e.g. 1.23. Equals the Amount divided by the Quantity.
Amount	Y	Float	1-7	Total amount for this line – this value would be equal to Units * Price.
/Item	Y	TAG	-	Closing tag.

Total	Y	Float	3-7	Total amount of the transaction – this value would be equal to the sum of all <*LineTotal*> elements sent.
/*TranDetail*	N	TAG	-	Closing tag.

Table 5

The <*TranDetail*> tag is used to send the transaction details generated within a transaction. When sending a Credit Sale transaction (CCSALE), the <*TranDetail*> tag and its associated elements can be sent to specify the full transaction details.

NOTE: The <*TranDetail*> is an optional element. However, when sent in a request message, some of its associated elements (marked as required in the table above), must also be included.

NOTE: There is no cross-checking or validation of data within the request message. For example, the <*Amount*> field and the <*Total*> field should equal the same amount, although the XML message is not checked/validated to ensure that this is the case.

A sample transaction with transaction details included:

```
<TranDetail>

      <ItemElement>
      <Item>Burger</Item>
      <Department>Food</Department>
      <SerialNo>0</SerialNo>
      <Quantity>2</Quantity>
      <PluNumber>173673613</PluNumber>
      <SerialNo>12345AB</SerialNo>
      <Price>2.50</Price>
      <Amount>5.00</Amount>
      </ItemElement>

      <ItemElement>
      <Item>Fry</Itemt>
      <Department>Food</Department>
      <SerialNo>0</SerialNo>
      <Quantity>1</Quantity>
      <PluNumber>173673613</PluNumber>
      <SerialNo>12345AB</SerialNo>
      <Price>5.99</Price>
```

```
      <Amount>5.99</Amount>
      </ItemElement>

      <ItemElement>
      <Item>Tax</Item>
      <Quantity>1</Quantity>
      <Amount>1</Amount>
      </ItemElement>

        <Total>17.97</Total>
</TranDetail>
```

Chapter 8: Transactions

Information pertaining to the transaction-specific elements is listed in the following sections.

Credit Card Transactions

Subsequent sections describe details of specific credit card transactions and their element tags. Common XML element tags for all credit card transactions (requests and responses), are outlined in the section that follows.

Common Elements for Requests

Tags common to all credit card request transactions, are outlined in the table below.

Element Tag	Required	Type	Size	Notes
Command	Y	Fixed String	-	Specifies transaction type to be executed Must be one of: CCPREAUTH/CCSALE/CCFINAL/ CCADJUSTTIP/CCVOID/CCREFUND
Id	N	String	1-20	Used by the POS system to track the transaction e.g. invoice # or receipt #. Sent unchanged in response.
ClientId	N	String	1-20	The ID of the user/cashier operating the PINPad. Sent unchanged in the response, and is not used by the POSLynx.
ClientMAC	Y- if requests are sent directly to POSLynx	String	12	MAC Address of client. This Id must be registered on POSLynx. This is added automatically by Precidia 'helper' applications (TNP-CG/TNP-NP), and required only when requests are sent directly to POSLynx.
IPAddress	N	String	7-15	Overrides the configured POSLynx IP Address. Allows controlling which POSLynx processes each transaction individually. Only applicable to TNP-CG
IPPort	N	Numeric	4-5	Overrides the configured POSLynx port. Used with *<IPAddress>* to control which POSLynx and port are used to process each transaction. Only applicable to TNP-CG
TaxAmount	N	Float	3-7	Amount of Tax in transaction. Used for Level 2 transaction reporting. Two decimal places. (e.g. 1.23) Only supported by some hosts.
CustomerCode	N	String	0-17	For Level 2 transaction reporting. Customers identifying information, such as Invoice or PO number. Only supported by some hosts currently.
TranDetail	N	TAG	-	See Section 7 - TranDetail can be used to send information about the transaction for detailed reporting.
Lane	N	String	1-20	Used to re-direct an XML message to a specific port within the same POSLynx, allowing an XML

				command to be sent to a non-XML Lane e.g. a cash register lane running an AT protocol. The value must match the lane name configured. Not returned in the response message. Currently supported for use with the following hosts:- Global, First Data, Paymentech, Precidia Gift Card server
Input	N	String	1-20	How the card was entered. One of SWIPED/MANUAL/EXTERNAL/TOKEN. See 'Card Entry Options' in Chapter 5 for more detail.

Table 6

Common Elements for Responses

Tags common to all credit card response transactions, are outlined in the table below.

Element Tag	Required	Type	Size	Notes
Result	Y	Fixed String	-	One of APPROVED / DECLINED / ERROR
Command	Y	Fixed String	-	Command returned is same as sent on the request.
ResultText	Y	String	1-30	Text to describe result – e.g. explain error or result that could be placed on POS screen.
Authorization	Y - If approved	String	1-20	Authorization code from processor.
RecNum	Y - If approved	String	4	Reference number used to refer to this transaction in future transactions (e.g. to void this transaction, RecNum will be used).
RefData	N	String	1-20	Additional reference data returned by host – may not be used – or may only be used by certain hosts.
ErrorCode	Y	String	4	Error code - 0000 if everything is successful. 1 if Result is DECLINED. Numeric error code if the Result is ERROR.
Id	N	String	1-20	Optional Id used by the POS system to track the transaction (e.g. invoice # or receipt #). Sent unchanged in response. If sent in request it will be returned in response.
ClientId	N	String	1-20	The ID of the user/cashier operating the PINPad. Sent unchanged in the response, and is not used by the POSLynx.
CardType	Y	Fixed String	-	Type of card sent in request. Visa/MasterCard/Amex/Diners/Discover/JCB/DEBIT/GIFT/CASH Based on card ranges defined in POSLynx configuration.
CardInfo	N	String	1-30	A representation of the card when passed through a unique one way hash algorithm that identifies the card, without allowing it to be reproduced. Allows e-receipts to be provided to customers based on a list of CARDINFO tags stored by the merchant. *Only available with EMV transactions.*
AuthAmt	Y – if Approved	Float	3-7	Amount that transaction was approved for.

CardNumber	Y	String	4	Returns last four digits of card number – for display if needed.
MerchantId	N	String	1-20	Merchant Id used to process transaction with host
TerminalId	N	String	1-20	Terminal Id used to process transaction with host
Receipt / Receipt#	N			If enabled in POSLynx, a printable host dependent Receipt will be part of the response. Not shown in examples, see section 6.
TransactionDate	N	Int	6	Date of Transaction – MMDDYY format. Not sent by all host configurations.
TransactionTime	N	Int	6	Time of Transaction – HHMMSS format. Not sent by all host configurations.
Token	N	String	16	A unique key value assigned to a PAN and used in future transactions in place of the PAN. Only supported by some Hosts.
InputMethod	N	String	1-20	How the card was entered. One of SWIPED/MANUAL/TOKEN/CHIP/TAP/RFID. See 'Card Entry Options' in Chapter 5 for more detail.

Table 7

Credit Sale Transaction

A Credit Sale transaction processes a payment on a credit card. The data can be swiped or manually entered. The Transaction is finalized on the card holder's credit card and funds are transferred to the merchant after batch close.

Request Elements

Please note that this list is not exhaustive. Other possible elements are covered in 'Common Elements' for Credit Card Requests.

Element Tag	Required	Type	Size	Notes
Command	Y	Fixed String	-	CCSALE
Amount	Y	Float	1-7	Total amount for this line – this value would be equal to Units * Price.
TaxAmount	N	Float	3-7	Amount of Tax in transaction. Used for Level 2 transaction reporting. Two decimal places. (e.g. 1.23) Only supported by some hosts.
Input	Y	Fixed String	-	Specifies type of card data given. Must be one of : SWIPED/MANUAL/EXTERNAL/TOKEN
Track2	N	String	1-40	Swiped track2 data - without start/end sentinel characters.
Track1	N	String	1-80	Swiped track1 data - without start/end sentinel characters.
CardNumber	Y - If Manual	Int	12-19	Account number. Required if MANUAL.
ExpiryDate	Y - If Manual	Int	4	Expiry Date in format MMYY. Required if MANUAL.
AVSStreet	N	String	1-40	AVS Street Address for verifying card user. Only for MANUAL transactions.
AVSZip	N	String	5-9	AVS Zip Code for verifying card user.Only for MANUAL transactions.
CVV	N	String	3-4	Card Verification Number for verifying card user. Called CVV2 by Visa,CVC2 by MC and CID by Amex.Only for MANUAL transactions In cases where the CVV is not available the following options should be provided:- 'A0' - CVV is deliberately bypassed or is not provided 'A2' - CVV is on the card but is illegible 'A9' - Cardholder states that card has no CVV imprint
IntervalPayment	N	Fixed String	-	Refers to transactions (or transaction sequences) that are processed at predefined intervals. One of Convenience, Installment, One_Time or Recurring. Only supported by certain hosts.

ConvenienceAmount	If IntervalPayment set to Convenience	Fixed String	-	The charge in addition to the original transaction amount for the convenience of being able to use an alternate payment method.
CommercialCard	N	Fixed String	-	Denotes whether or not the card in question is a commercial card.
EncryptionType	Y - if using Encryption Card Readers	Fixed String	Magtek V1 or Magtek V2	Details the type of Encryption Card Reader.
ECD	Y - if using Encryption Card Readers	String		The encrypted data read from the card.

Table 8

Example Request

A swiped transaction:

```
<PLRequest>
    <Command>CCSALE</Command>
    <Id>9999</Id>
    <Amount>55.45</Amount>
    <TaxAmount>1.12</TaxAmount>
    <Input>SWIPED</Input>
    <Track2>5454545454545454=101220134350543</Track2>
    <Track1>5454545454545454^LASTNAME/FIRSTNAME^10122013430543312410</Track1>
    <IPAddress>12.34.56.78</IPAddress>
    <IPPort>12345</IPPort>
    <CustomerCode>9876ABCD</CustomerCode>

</PLRequest>
```

A manually keyed transaction:

```
<PLRequest>
    <Command>CCSALE</Command>
    <Id>9999</Id>
    <Amount>55.45</Amount>
    <Input>MANUAL</Input>
    <CardNumber>5454545454545454</CardNumber>
    <ExpiryDate>0114</ExpiryDate>
    <AVSStreet>123 Main</AVSStreet>
    <AVSZip>90210</AVSZip>
    <CVV>987</CVV>

</PLRequest>
```

Response Elements

Please note that this list is not exhaustive. Other possible elements are covered in 'Common Elements' for Credit Card Responses.

Element Tag	Required	Type	Size	Notes
CVVResult	N	String	1-20	Result returned by host for CVV data entered. Only returned for manually entered transactions. Host specific.
AVSZipResult	N	String	1-20	Result returned by host for AVS Zip data entered. Only returned for manually entered transactions. Host specific.
AVSStreetResult	N	String	1-20	Result returned by host for AVS Street data entered. Only returned for manually entered transactions. Host specific.
CustomerName	N	String	1-20	Customer name found in Track1 input. Only returned if Track1 data sent in request.
InputMethod	N	String	1-20	How the card was entered. One of SWIPED/MANUAL/TOKEN/CHIP/TAP/RFID. See 'Card Entry Options' in Chapter 5 for more detail.

Table 9

Example Response

An approved transaction response:

```
<PLResponse>
    <Result>APPROVED</Result>
    <Command>CCSALE</Command>
    <Id>9999</Id>
    <ResultText>AP</ResultText>
    <Authorization>123456</Authorization>
    <AuthAmt>55.45</AuthAmt>
    <CardNumber>5454</CardNumber>
    <RefData>0000</RefData>
    <RecNum>2001</RecNum>
    <TerminalId>001</TerminalId>
    <ErrorCode>0000</ErrorCode>
    <CardType>MasterCard</CardType>
    <CardInfo>5B0EC671F5B9E…B3208A84310CB615ED56A991DA3A6E018</CardInfo>
    <TransactionDate>010311</TransactionDate>
    <TransactionTime>144048</TransactionTime>
    <InputMethod>SWIPED</InputMethod>
</PLResponse>
```

A declined transaction response (invalid expiry date):

```
<PLResponse>
    <Result>DECLINED</Result>
    <ErrorCode>0001</ErrorCode>
    <Command>CCSALE </Command>
    <Id>9999</Id>
    <ResultText>Bad Expiry Date</ResultText>
    <RefData>0000</RefData>
    <RecNum>2001</RecNum>
    <TerminalId>001</TerminalId>
    <MerchantId>1234567890</MerchantId>
    <CardNumber>5454</CardNumber>
    <CardType>MasterCard</CardType>
    <CardInfo>F710F77E85D3450DA2…66EB757CF72F9C9E723A7FC2D336</CardInfo>
    <InputMethod>MANUAL</InputMethod>
</PLResponse>
```

Credit Sale Using Tokens

The use of Tokens is a two-step process. First, a transaction takes place, whereby the merchant will use the card number or track2 data. In the response to this transaction request, a Token is included which can be used in subsequent transactions in place of the card-holder's credit-card data.

Token Format (Merchant Link)

An example of a Merchant Link (ML) Token is shown.

8049529970**1**1**4444**

The format of the ML Token is defined by the following:

- All Tokens are 16 digits in length
- The first digit (left most significant) of the token will always be an 8
- The second and third digits (from left most significant) identify the Card Type.

  Default values for Card Type identifiers are:

  34 = Amex

  44 = Visa

  54 = MasterCard

  64 = Discover (Diner's and JCB)

- Eight of the middle nine digits are generated from the Token Vault.

- The 11th digit is a check to pass the MOD-10 Luhn check.
- The last 4 digits are the last four digits of the actual Card Number

NOTE: The same Token will be returned each time the Card Number is presented to Merchant Link. ML Tokens are not tied to the card's Expiration Date and will not expire.

NOTE: For ML Tokens that pass the MOD-10 requirement, there will be approximately 9 billion unique Tokens per Card Type.

CCSale Request—Initial Transaction (No Token)

The initial CCSale request is sent without Tokens:

```
<PLRequest>
      <Command>CCSALE</Command>
      <Input>SWIPED</Input>
      <Id>9999</Id>
      <Track2>4485581000000005=151210100000832</Track2>
      <Amount>5.00</Amount>
</PLRequest>
```

CCSale Response—Token Received in Response

The response included a *<Token>* element, which assigns the value of the Token that can be used in subsequent transactions, as shown below:

```
<PLResponse>
      <Result>APPROVED</Result>
      <Authorization />
      <RecNum>000680</RecNum>
      <RefData>MV0002475260</RefData>
      <Language>English</Language>
      <ErrorCode>0000</ErrorCode>
      <AuthAmt>5.00</AuthAmt>
      <Token>8440000188910005</Token>
      <CardNumber>0005</CardNumber>
      <CardType>VISA</CardType>
      <CardInfo>7E751AFCA3168C125F351826238F5C6A6C80ABDB8AD961F9E7723020FEABC
       A92F7B1A9391FFF5CC8EFEF7321818FCB0C2C1BD898AFEA4CAF71</CardInfo>
```

```
      <TransactionDate>130516</TransactionDate>
      <TransactionTime>151129</TransactionTime>
      <Command>CCSALE</Command>
      <Id>9999</Id>
      <ResultText>APPROVED</ResultText>
      <MerchantId>PRECIDIA1111111</MerchantId>
      <TerminalId>111111</TerminalId>
      <InputMethod>SWIPED</InputMethod>
</PLResponse>
```

CCSale Request Using Token

In the subsequent request, the *<Input>* element is set to 'TOKEN' and the *<Token>* element is included in the message along with the Token value that was provided in the previous CCSale response message.

The CCSale request using Token is shown below:

```
<PLRequest>
      <Command>CCSALE</Command>
      <Input>TOKEN</Input>
      <Id>9999</Id>
      <Token>8440000188910005</Token>
      <ExpiryDate>1215</ExpiryDate>
      <Amount>1.50</Amount>
</PLRequest>
```

CCSale Response Using Token

The CCSale response using Token is shown below:

```
<PLResponse>
      <Result>APPROVED</Result>
      <Authorization />
      <RecNum>000681</RecNum>
      <Id>9999</Id>
      <RefData>MV0002475262</RefData>
      <Language>English</Language>
      <ErrorCode>0000</ErrorCode>
      <AuthAmt>1.50</AuthAmt>
      <Token>8440000188910005</Token>
      <TransactionDate>130516</TransactionDate>
      <TransactionTime>151446</TransactionTime>
      <Command>CCSALE</Command>
```

```
        <ResultText>APPROVED</ResultText>
        <MerchantId>PRECIDIA1111111</MerchantId>
        <TerminalId>111111</TerminalId>
        <InputMethod>TOKEN</InputMethod>
</PLResponse>
```

Point-to-Point Encryption (P2PE)

> **NOTE:** This feature is currently only supported with Merchant Link transactions.

With the introduction of encryption card-readers, it is possible to conduct a transaction using encrypted card data from the entry point of a merchant's point-of-sale device to the point of secure decryption by the payment processor. Currently, only Magtek Encryption readers are supported.

When sending a transaction using the TNP-CG with the Input element set to Manual (required when Input and Track data are not provided), a prompt will ask for the card to be swiped. Following the swipe, two additional elements (EncryptionType and ECD) are added to the messages and passed to the POSLynx.

Configuring TNP-CG for P2PE

Make the following changes to the TNP-CG's settings under the **Peripheral and GUI** tab:

1. Select either *Magtek V1* or *Magtek V2* (*Magtek V2* is most common) from the local **POS Device** drop-down menu.
2. Select *<Keyboard Emulation>* from the **Connection** drop-down menu.
3. Verify the **Use GUI Prompts** checkbox is checked.
4. Click *<OK>.*

CCSale Request to TNP-CG

The initial CCSale request:

```
<PLRequest>
        <Command>CCSALE</Command>
        <Input>MANUAL</Input>
        <Id>9999</Id>
        <Amount>10.25</Amount>
</PLRequest>
```

CCSale Request (if sent directly to POSLynx)

Please note that the EncryptionType and ECD fields are automatically added to the request.

```
<PLRequest>
      <Command>CCSALE</Command>
      <Amount>10.25</Amount>
      <Id>9999</Id>
      <EncryptionType>MagtekV2</EncryptionType>
</PLRequest>
```

Pre-Authorize Transaction

This command is used to authorize a transaction without actually placing a charge on a card. It ensures there is credit available for the transaction when finalized (that is, puts a temporary hold on the funds for the card holder).

Can be swiped or manually keyed.

Request Elements

The following table outlines the element tags.

Element Tag	Required	Type	Size	Notes
Command	Y	Fixed String	-	CCPREAUTH
AuthAmt	Y	Float	3-7	Amount to be authorized for this transaction. POS may include an estimated gratuity in this amount to ensure that finalized transaction will be approved. Two decimal places, no dollar sign – (e.g. 123.45) Positive values only.
Input	Y	Fixed String	-	Specifies type of card data given. Must be one of : SWIPED/MANUAL/EXTERNAL/TOKEN
Track2	Y - If Swiped	String	1-40	Swiped track2 data - without sentinel characters. Required if SWIPED
Track1	N	String	1-80	Swiped track1 data - without start/end sentinel characters.
CardNumber	Y - If Manual	Int	12-19	Account number. Required if MANUAL.
ExpiryDate	Y - If Manual	Int	4	Expiry Date in format MMYY. Required if MANUAL.
AVSStreet	N	String	1-40	AVS Street Address for verifying card user. Only for MANUAL transactions.
AVSZip	N	String	5-9	AVS Zip Code for verifying card user. Only for MANUAL transactions.
CVV	N	Int	3-4	Card Verification Number for verifying card user. Called CVV2 by Visa, CVC2 by MC and CID by Amex. Only for MANUAL transactions.

Table 10

Example Request

A swiped transaction:

```
<PLRequest>
    <Command>CCPREAUTH</Command>
    <AuthAmt>30.00</AuthAmt>
    <Id>9999</Id>
    <Input>SWIPED</Input>
    <Track2>5454545454545454=101220134350543</Track2>
```

```
    <Track1>5454545454545454^LASTNAME/FIRSTNAME^10122013435054331l0</Track1>
    <IPAddress>12.34.56.78</IPAddress>
    <IPPort>12345</IPPort>
</PLRequest>
```

A manually keyed transaction:

```
<PLRequest>
    <Command>CCPREAUTH</Command>
    <AuthAmt>30.00</AuthAmt>
    <Id>9999</Id>
    <Input>MANUAL</Input>
    <CardNumber>5454545454545454</CardNumber>
    <ExpiryDate>0114</ExpiryDate>
    <AVSStreet>123 Main</AVSStreet>
    <AVSZip>90210</AVSZip>
    <CVV>987</CVV>
</PLRequest>
```

Response Elements

Please note that this list is not exhaustive. Other possible elements are covered in 'Common Elements' for Credit Card Responses.

Element Tag	Required	Type	Size	Notes
CVVResult	N	String	1-20	Result returned by host for CVV data entered. Only returned for manually entered transactions. Host specific.
AVSZipResult	N	String	1-20	Result returned by host for AVS Zip data entered. Only returned for manually entered transactions. Host specific.
AVSStreetResult	N	String	1-20	Result returned by host for AVS Street data entered. Only returned for manually entered transactions. Host specific.
CustomerName	N	String	1-20	Customer name found in Track1 input. Only returned if Track1 data sent in request.
InputMethod	N	String	1-20	How the card was entered. One of SWIPED/MANUAL/TOKEN/CHIP/TAP/RFID. See 'Card Entry Options' in Chapter 5 for more detail.

Table 11

Example Response

An approved transaction response:

```
<PLResponse>
    <Result>APPROVED</Result>
    <Command>CCPREAUTH</Command>
    <Id>9999</Id>
    <ResultText>Transaction Approved</ResultText>
    <Authorization>123456</Authorization>
    <RefData>0000</RefData>
    <RecNum>2001</RecNum>
    <TerminalId>001</TerminalId>
    <MerchantId>1234567890</MerchantId>
    <ErrorCode>0000</ErrorCode>
    <CardType>MasterCard</CardType>
    <InputMethod>MANUAL</InputMethod>
</PLResponse>
```

A declined transaction response:

```
<PLResponse>
    <Result>DECLINED</Result>
    <Command>CCPREAUTH</Command>
    <Id>9999</Id>
    <ResultText>Bad Expiry Date</ResultText>
    <RefData>0000</RefData>
    <RecNum>2001</RecNum>
    <TerminalId>001</TerminalId>
    <MerchantId>1234567890</MerchantId>
    <ErrorCode>0100</ErrorCode>
    <CardType>MasterCard</CardType>
    <InputMethod>MANUAL</InputMethod>
</PLResponse>
```

Pre-Auth Completion Transaction

This type finalizes the previously approved pre-authorization request. The transaction will be placed on the customer card and when the batch is closed the funds will be transferred to the merchant. If this is not done, then the Pre Auth transaction will eventually timeout and no transaction will take place.

This transaction is tied to the previous transaction using the <*RecNum*>.

Request Elements

The following table outlines the associated tags. Please note that this list is not exhaustive. Other possible elements are covered in 'Common Elements' for Credit Card Requests.

Element Tag	Required	Type	Size	Notes
Command	Y	Fixed String	-	CCFINAL
Amount	Y	Float	1-7	Actual Purchase Amount of this transaction. Two decimal places, no dollar sign – (e.g. 123.45). Positive values only.
Gratuity	N	Float	3-7	Amount of gratuity for this transaction. Total transaction amount will be for total of Amount+Gratuity. Positive values only.
AuthAmt	Y	Float	3-7	Full dollar Amount of this transaction (Amount+Gratuity). Two decimal places, no dollar sign – (e.g. 123.45). Positive values only.
RecNum	Y	String	4	Reference to the original CCPREAUTH transaction.
Input	Y	String	1-20	How the card was entered. One of SWIPED/MANUAL/EXTERNAL/TOKEN. See 'Card Entry Options' in Chapter 5 for more detail.

Table 12

Example Request

```
<PLRequest>
    <Command>CCFINAL</Command>
    <Amount>25.00</Amount>
    <Id>9999</Id>
    <Gratuity>2.50</Gratuity>
    <AuthAmt>27.50</AuthAmt>
    <RecNum>2001</RecNum>
    <Input>SWIPED</Input>
</PLRequest>
```

Response Elements

Please note that this list is not exhaustive. Other possible elements are covered in 'Common Elements' for Credit Card Responses.

Element Tag	Required	Type	Size	Notes
Command	Y	Fixed String	-	CCFINAL
Amount	Y	Float	1-7	Actual Purchase Amount of this transaction. Two decimal places, no dollar sign – (e.g. 123.45). Positive values only.
Gratuity	N	Float	3-7	Amount of gratuity for this transaction. Total transaction amount will be for total of Amount+Gratuity. Positive values only.
AuthAmt	Y	Float	3-7	Full dollar Amount of this transaction (Amount+Gratuity). Two decimal places, no dollar sign – (e.g. 123.45). Positive values only.
RecNum	Y	String	4	Reference to the original CCPREAUTH transaction.
InputMethod	N	String	1-20	How the card was entered. One of SWIPED/MANUAL/TOKEN/CHIP/TAP/RFID. See 'Card Entry Options' in Chapter 5 for more detail.

Table 13

Example Response

```
<PLResponse>
    <Result>APPROVED</Result>
    <Command>CCFINAL </Command>
    <Id>9999</Id>
    <ResultText>Transaction Approved</ResultText>
    <Authorization>123456</Authorization>
    <RefData>0000</RefData>
    <RecNum>2001</RecNum>
    <TerminalId>001</TerminalId>
    <MerchantId>1234567890</MerchantId>
    <ErrorCode>0000</ErrorCode>
    <InputMethod>SWIPED</InputMethod>
</PLResponse>
```

Re-Authorize Transaction

This type is used to re-authorize a previously authorized transaction. When the PREAUTH transaction is performed, the authorization is usually valid for several days (the exact time is processor dependent). However, on occasion, the authorization may be held longer (for example, when a mail order shipment is not ready).

Performing this transaction will clear the original authorization and create a new one and tends to be expensive as multiple transactions are done on the host. This transaction also allows a new amount to be authorized.

Initially the original record number is sent in the request, although a new record number will be returned, which must subsequently be used to finalize the transaction.

Request Elements

Please note that this list is not exhaustive. Other possible elements are covered in 'Common Elements' for Credit Card Requests.

Element Tag	Required	Type	Size	Notes
Command	Y	Fixed String	-	CCREAUTH
AuthAmt	Y	Float	3-7	Amount to be authorized for this transaction. POS may include an estimated gratuity in this amount to ensure that finalized transaction will be approved. Two decimal places, no dollar sign – (e.g. 123.45). Positive values only.
RecNum	Y	String	4	Reference to the original Preauth transaction.
Input	Y	String	1-20	How the card was entered. One of SWIPED/MANUAL/EXTERNAL/TOKEN. See 'Card Entry Options' in Chapter 5 for more detail.

Table 14

Example Request

```
<PLRequest>
    <Command>CCREAUTH</Command>
    <Id>9999</Id>
    <AuthAmt>30.00</AuthAmt>
    <RecNum>123</RecNum>
    <Input>SWIPED</Input>
</PLRequest>
```

Response Elements

Please note that this list is not exhaustive. Other possible elements are covered in 'Common Elements' for Credit Card Responses.

Element Tag	Required	Type	Size	Notes
Command	Y	Fixed String	-	CCREAUTH
Id	N	String	1-20	Optional Id used by the POS system to track the transaction (e.g. invoice # or receipt #). Sent unchanged in response. If sent in request it will be returned in response.
TerminalID	N	String	1-20	Merchant Id used to process transaction with host
MerchantID	N	String	1-20	Terminal Id used to process transaction with host
RefData	N	String	1-20	Additional reference data returned by host – may not be used – or may only be used by certain hosts.
RecNum	Y	String	4	Reference to the original CCPREAUTH transaction.
InputMethod	N	String	1-20	How the card was entered. One of SWIPED/MANUAL/TOKEN/CHIP/TAP/RFID. See 'Card Entry Options' in Chapter 5 for more detail.

Table 15

Example Response

```
<PLResponse>
    <Result>APPROVED</Result>
    <Command>CCREAUTH</Command>
    <Id>9999</Id>
    <ResultText>Transaction Approved</ResultText>
    <Authorization>123456</Authorization>
    <RefData>0000</RefData>
    <RecNum>128</RecNum>
    <TerminalId>001</TerminalId>
    <MerchantId>1234567890</MerchantId>
    <ErrorCode>0000</ErrorCode>
    <InputMethod>SWIPED</InputMethod>
</PLResponse>
```

Refund Transaction

This transaction credits a customer account. In most cases, it is used to return money back to the customer when purchased goods are returned.

This is not tied to any previous transaction and can be swiped or manually entered.

Request Elements

The following table outlines the associated tags. Please note that this list is not exhaustive. Other possible elements are covered in 'Common Elements' for Credit Card Requests.

Element Tag	Required	Type	Size	Notes
Command	Y	Fixed String	-	CCREFUND
Amount	Y	Float	1-7	Amount to be refunded. Two decimal places, no dollar sign – (e.g. 123.45) Positive values only.
Input	Y	Fixed String	-	Specifies type of card data given. Must be one of : SWIPED/MANUAL/EXTERNAL
Track2	Y - If Swiped	String	1-40	Swiped track2 data - without sentinel characters.Required if SWIPED
Track1	N	String	1-80	Swiped track1 data - without start/end sentinel characters.
CardNumber	Y - If Manual	Int	12-19	Account number. Required if MANUAL.
ExpiryDate	Y - If Manual	Int	4	Expiry Date in format MMYY. Required if MANUAL.
AVSStreet	N	String	1-40	AVS Street Address for verifying card user. Only for MANUAL transactions.

AVSZip	N	String	5-9	AVS Zip Code for verifying card user. Only for MANUAL transactions.
CVV	N	Int	3-4	Card Verification Number for verifying card user. Called CVV2 by Visa, CVC2 by MC and CID by Amex. Only for MANUAL transactions.

Table 16

Example Request

```
<PLRequest>
    <Command>CCREFUND</Command>
    <Id>9999</Id>
    <Amount>15.00</Amount>
    <Input>SWIPED</Input>
    <Track2>5454545454545454=101220134350543</Track2>
    <Track1>5454545454545454^LASTNAME/FIRSTNAME^101220134350543310</Track1>
    </PLRequest>
```

NOTE: If a card is manually entered, *<Track1>/<Track2>* elements would not be specified.

```
    <Input>MANUAL</Input>
    <CardNumber>5454545454545454</CardNumber>
    <ExpiryDate>0114</ExpiryDate>
```

Additional verification information can also be included for a manual transaction.

```
    <AVSStreet>123 Main</AVSStreet>
    <AVSZip>90210</AVSZip>
    <CVV>987</CVV>
```

Response Elements

Please note that this list is not exhaustive. Other possible elements are covered in 'Common Elements' for Credit Card Responses.

Element Tag	Required	Type	Size	Notes
CVVResult	N	String	1-20	Result returned by host for CVV data entered.Only returned for manually entered transactions. Host specific.
AVSZipResult	N	String	1-20	Result returned by host for AVS Zip data entered.Only returned for manually entered transactions. Host specific.
AVSStreetResult	N	String	1-20	Result returned by host for AVS Street data entered. Only returned for manually entered

				transactions. Host specific.
CustomerName	N	String	1-20	Customer name found in Track1 input. Only returned if Track1 data sent in request.
InputMethod	N	String	1-20	How the card was entered. One of SWIPED/MANUAL/TOKEN/CHIP/TAP/RFID. See 'Card Entry Options' in Chapter 5 for more detail.

Table 17

Example Response

```
<PLResponse>
    <Result>APPROVED</Result>
    <Command>CCREFUND</Command>
    <Id>9999</Id>
    <ResultText>Transaction Approved</ResultText>
    <Authorization>987654</Authorization>
    <RefData>111222333</RefData>
    <RecNum>2003</RecNum>
    <TerminalId>001</TerminalId>
    <MerchantId>1234567890</MerchantId>
    <ErrorCode>0000</ErrorCode>
    <CardType>MasterCard</CardType>
    <InputMethod>SWIPED</InputMethod>
</PLResponse>
```

Void Transaction

This transaction type removes a previous transaction from the merchant batch. That is, it cancels the transaction. A Void transaction is used to remove a charge from a customer when a problem is encountered with the transaction. If the batch has been closed, a Void transaction cannot be performed. In this case, a Refund transaction must be made.

This transaction is tied to the previous transaction via the *<RecNum>* element.

Request Elements

Please note that this list is not exhaustive. Other possible elements are covered in 'Common Elements' for Credit Card Requests.

Element Tag	Required	Type	Size	Notes
Command	Y	Fixed String	-	CCVOID
Amount	Y	Float	1-7	Amount of original transaction. Two decimal places, no dollar sign – (e.g. 123.45) Positive values only. Used as extra confirmation that correct transaction is voided.
RecNum	Y	String	4	Reference to the original transaction to be voided.
Input	Y	String	1-20	How the card was entered. One of SWIPED/MANUAL/EXTERNAL/TOKEN. See 'Card Entry Options' in Chapter 5 for more detail.

Table 18

Example Request

```
<PLRequest>
    <Command>CCVOID</Command>
    <Id>9999</Id>
    <Amount>15.00</Amount>
    <RecNum>2001</RecNum>
    <Input>SWIPED</Input>
</PLRequest>
```

Response Elements

Please note that this list is not exhaustive. Other possible elements are covered in 'Common Elements' for Credit Card Responses.

Element Tag	Required	Type	Size	Notes
ResultText	Y	String	1-30	Text to describe result – e.g. explain error or result that could be placed on POS screen.
Authorization	Y - If approved	String	1-20	Authorization code from processor.
RecNum	Y - If	String	4	Reference number used to refer to this transaction

	approved			in future transactions (e.g. to void this transaction, RecNum will be used).
RefData	N	String	1-20	Additional reference data returned by host – may not be used – or may only be used by certain hosts.
ErrorCode	Y	String	4	Error code - 0000 if everything is successful. 1 if Result is DECLINED. Numeric error code if the Result is ERROR.
Id	N	String	1-20	Optional Id used by the POS system to track the transaction (e.g. invoice # or receipt #). Sent unchanged in response. If sent in request it will be returned in response.
InputMethod	N	String	1-20	How the card was entered. One of SWIPED/MANUAL/TOKEN/CHIP/TAP/RFID. See 'Card Entry Options' in Chapter 5 for more detail.
MerchantId	N	String	1-20	Merchant Id used to process transaction with host
TerminalId	N	String	1-20	Terminal Id used to process transaction with host

Example Response

```
<PLResponse>
    <Result>APPROVED</Result>
    <Command>CCVOID</Command>
    <Id>9999</Id>
    <ResultText>Transaction Approved</ResultText>
    <Authorization>111111</Authorization>
    <RefData>0000</RefData>
    <RecNum>2001</RecNum>
    <TerminalId>001</TerminalId>
    <MerchantId>1234567890</MerchantId>
    <ErrorCode>0000</ErrorCode>
    <InputMethod>SWIPED</InputMethod>
</PLResponse>
```

Voice Authorization/Force Transaction

When a system is down, it may be necessary to manually obtain telephone credit card approval from the Card Authorization Center. The Force Transaction command will place the transaction into the batch.

Request Elements

Please note that this list is not exhaustive. Other possible elements are covered in 'Common Elements' for Credit Card Requests.

Element Tag	Required	Type	Size	Notes
Command	Y	Fixed	-	CCVOICE

		String		
Amount	Y	Float	1-7	Amount of transaction. Two decimal places, no dollar sign – (e.g. 123.45) Positive values only.
VoiceAuth	Y	String	6	Authorization code obtained via phone approval.
Input	Y	Fixed String	-	Specifies type of card data given. Must be one of : SWIPED/MANUAL/EXTERNAL
Track2	Y - If Swiped	String	1-40	Swiped track2 data - without sentinel characters. Required if SWIPED
Track1	N	String	1-80	Swiped track1 data - without start/end sentinel characters.
CardNumber	Y - If Manual	Int	12-19	Account number. Required if MANUAL.
ExpiryDate	Y - If Manual	Int	4	Expiry Date in format MMYY. Required if MANUAL.
AVSStreet	N	String	1-40	AVS Street Address for verifying card user. Only for MANUAL transactions.
AVSZip	N	String	5-9	AVS Zip Code for verifying card user. Only for MANUAL transactions.
CVV	N	Int	3-4	Card Verification Number for verifying card user. Called CVV2 by Visa, CVC2 by MC and CID by Amex. Only for MANUAL transactions.
RecNum	N for most hosts, Y for First Data	String	4	Only used in CCVOICE for First Data host. Reference to the original transaction.

Table 19

Example Request

```
<PLRequest>
    <Command>CCVOICE</Command>
    <Id>9999</Id>
    <Amount>55.45</Amount>
    <VoiceAuth>123987</VoiceAuth>
    <Input>SWIPED</Input>
    <Track2>5454545454545454=101220134350543</Track2>
    <Track1>5454545454545454^LASTNAME/FIRSTNAME^10122013435052410</Track1>
</PLRequest>
```

NOTE: Should the card be manually entered, *<Track1>/<Track2>* elements will be replaced by the *<CardNumber>* and *<ExpiryDate>* elements, as follows below.

```
    <Input>MANUAL</Input>
    <CardNumber>5454545454545454</CardNumber>
    <ExpiryDate>0114</ExpiryDate>
```

Additional verification information can also be included for manual transactions.

```
<AVSStreet>123 Main</AVSStreet>
<AVSZip>90210</AVSZip>
<CVV>987</CVV>
```

Response Elements

Please note that this list is not exhaustive. Other possible elements are covered in 'Common Elements' for Credit Card Responses.

Element Tag	Required	Type	Size	Notes
CVVResult	N	String	1-20	Result returned by host for CVV data entered. Only returned for manually entered transactions. Host specific.
AVSZipResult	N	String	1-20	Result returned by host for AVS Zip data entered. Only returned for manually entered transactions. Host specific.
AVSStreetResult	N	String	1-20	Result returned by host for AVS Street data entered. Only returned for manually entered transactions. Host specific.
InputMethod	N	String	1-20	How the card was entered. One of SWIPED/MANUAL/TOKEN/CHIP/TAP/RFID. See 'Card Entry Options' in Chapter 5 for more detail.
CustomerName	N	String	1-20	Customer name found in Track1 input. Only returned if Track1 data sent in request.

Table 20

Example Response

```
<PLResponse>
    <Result>APPROVED</Result>
    <Command>CCVOICE</Command>
    <Id>9999</Id>
    <ResultText>Transaction Approved</ResultText>
    <Authorization>123456</Authorization>
    <RefData>0000</RefData>
    <RecNum>0005</RecNum>
    <TerminalId>001</TerminalId>
    <MerchantId>1234567890</MerchantId>
    <ErrorCode>0</ErrorCode>
    <CardType>MasterCard</CardType>
    <InputMethod>SWIPED</InputMethod>
```

```
</PLResponse>
```

Adjust Tip Transaction

This transaction type modifies the Tip amount of a previously completed credit card transaction.

This transaction is tied to the previous transaction via the *<RecNum>* and the original base amount.

Request Elements

Please note that this list is not exhaustive. Other possible elements are covered in 'Common Elements' for Credit Card Requests.

Element Tag	Required	Type	Size	Notes
Command	Y	Fixed String	-	CCADJUSTTIP
Id	N	String	1-20	Used by the POS system to track the transaction e.g. invoice # or receipt #. Sent unchanged in response.
Amount	Y	Float	1-7	Base Purchase Amount of original transaction. Two decimal places, no dollar sign – (e.g. 123.45). Positive values only. Used as extra confirmation that correct transaction is adjusted.
Gratuity	Y	Float	3-7	New Amount of gratuity for this transaction. Total transaction amount will be for total of Amount+Gratuity. Two decimal places, no dollar sign – (e.g. 123.45). Positive values only.
RecNum	Y	String	4	Reference to the original transaction to be voided.

Table 21

Example Request

```
<PLRequest>
    <Command>CCADJUSTTIP</Command>
    <Id>9999</Id>
    <Amount>15.00</Amount>
    <Gratuity>1.50</Gratuity>
    <RecNum>12</RecNum>
    <Input>SWIPED</Input>
</PLRequest>
```

Response Elements

Please note that this list is not exhaustive. Other possible elements are covered in 'Common Elements' for Credit Card Responses.

Element Tag	Required	Type	Size	Notes
InputMethod	N	String	1-20	How the card was entered. One of SWIPED/MANUAL/TOKEN/CHIP/TAP/RFID. See 'Card Entry Options' in Chapter 5 for more detail.
ResultText	Y	String	1-30	Text to describe result – e.g. explain error or result that could be placed on POS screen.
Authorization	Y - If approved	String	1-20	Authorization code from processor.
RecNum	Y - If approved	String	4	Reference number used to refer to this transaction in future transactions (e.g. to void this transaction, RecNum will be used).
RefData	N	String	1-20	Additional reference data returned by host – may not be used – or may only be used by certain hosts.
ErrorCode	Y	String	4	Error code - 0000 if everything is successful. 1 if Result is DECLINED. Numeric error code if the Result is ERROR.
Id	N	String	1-20	Optional Id used by the POS system to track the transaction (e.g. invoice # or receipt #). Sent unchanged in response. If sent in request it will be returned in response.
InputMethod	N	String	1-20	How the card was entered. One of SWIPED/MANUAL/TOKEN/CHIP/TAP/RFID. See 'Card Entry Options' in Chapter 5 for more detail.
MerchantId	N	String	1-20	Merchant Id used to process transaction with host
TerminalId	N	String	1-20	Terminal Id used to process transaction with host

Example Response

```
<PLResponse>
    <Result>APPROVED</Result>
    <Command>CCADJUSTTIP</Command>
    <Id>9999</Id>
    <ResultText>Transaction Approved</ResultText>
    <Authorization>111222</Authorization>
    <RefData>0000</RefData>
    <RecNum>0012</RecNum>
    <TerminalId>001</TerminalId>
    <MerchantId>1234567890</MerchantId>
    <ErrorCode>0000</ErrorCode>
    <InputMethod>SWIPED</InputMethod>
</PLResponse>
```

Optional First Data BUYPASS Fuel

This credit card transaction type performs additions to only support First Data BUYPASS fuel cards. The following table outlines the associated tags.

Request Elements

Please note that this list is not exhaustive. Other possible elements are covered in 'Common Elements' for Credit Card Requests.

Element Tag	Required	Type	Size	Notes
Command	Y	Fixed String	-	CCSALE for fuel cards handled by FirstData BUYPASS ONLY
Amount	Y	Float	1-7	Amount of transaction. Two decimal places, no dollar sign – (e.g. 123.45) Positive values only.
FuelAmount	Y	Float	3-7	The amount of fuel and/or maintenance depending on the restrictions of the card. If non-fuel amount is 0.00 then an amount of 0.00 is still required
FuelCostPerUnit	Y	Float	3-7	The cost per gallon of the fuel.
ProductCodeInfo	C	String	1-20	The contents of the Product Code Data Segment, as described on page 14-8 of the BUYPASS Platform ATL105 Message Format Specifications, version 2011-2. The Segment header (elements 84 and 85) and file separator are not required.
Odometer	C	String	1-20	The Odometer reading from the vehicle
VehicleID	C	String	1-20	The vehicle ID
VehicleNum	C	String	1-20	The vehicle Number
DriverID	C	String	1-20	The driver identification number
PumpLaneNum	C	Integer	2	The pump or lane number of the fuel pump
Input	N	String	1-20	How the card was entered. One of SWIPED/MANUAL/EXTERNAL/TOKEN. See 'Card Entry Options' in Chapter 5 for more detail.
Id	N	String	1-20	Used by the POS system to track the transaction e.g. invoice # or receipt #. Sent unchanged in response.

Table 22

BUYPASS requires fuel charges to be listed separately from other purchases - the *<FuelAmount>* tag is used to identify this amount.

Also worthy of note is the *<ProductCodeInfo>* element, which is used to create the 102 segments required for fuel purchases. The ECR must create the segments and the POSLynx software will add the headers and send in the complete segment (along with any other required segments).

Example Request

```
<PLRequest>
    <Command>CCSALE</Command>
```

```
    <Amount>9.00</Amount>
    <Id>9999</Id>
    <FuelAmount>70.00</FuelAmount>
    <FuelCostPerUnit>3.20</FuelCostPerUnit>
    <ProductCodeInfo>TBD</ProductCodeInfo>
    <Odometer>1999012</Odometer>
    <VehicleID>1111</VehicleID>
    <PumpLaneNum>99</PumpLaneNum>
    <Input>SWIPED</Input>
</PLRequest>
```

Example Response

```
<PLResponse>
    <Result>APPROVED</Result>
    <Command>CCSALE</Command>
    <ResultText>Transaction Approved</ResultText>
    <Authorization>111222</Authorization>
    <Id>9999</Id>
    <RefData>0000</RefData>
    <RecNum>0012</RecNum>
    <TerminalId>001</TerminalId>
    <MerchantId>1234567890</MerchantId>
    <AuthAmt>40.00</AuthAmt>
    <FuelAmount>40.00</FuelAmount>
    <FuelCostPerUnit>3.20</FuelCostPerUnit>
    <PumpLaneNum>99</PumpLaneNum>
    <ErrorCode>0000</ErrorCode>
    <InputMethod>SWIPED</InputMethod>
</PLResponse>
```

Example Request

```
<PLRequest>
    <Command>CCSALE</Command>
    <Amount>0.00</Amount>
    <Id>9999</Id>
    <FuelAmount>10.00</FuelAmount>
    <FuelCostPerUnit>1.00</FuelCostPerUnit>
    <PumpLaneNum>2</PumpLaneNum>
    <Track2>70764912345100039=9912</Track2>
```

```
    <DriverID>11411</DriverID>
    <ClientMAC>FFFFFFFFFFFF</ClientMAC>
    <ProductCodeInfo>F01020G010000\0010000\100</ProductCodeInfo>
    <Input>SWIPED</Input>
</PLRequest>
```

Example Request

```
<PLRequest>
    <Command>CCSALE</Command>
    <Amount>5.00</Amount>
    <Id>9999</Id>
    <FuelAmount>0.00</FuelAmount>
    <FuelCostPerUnit>1.00</FuelCostPerUnit>
    <PumpLaneNum>2</PumpLaneNum>
    <DriverID>11411</DriverID>
    <ClientMAC>FFFFFFFFFFFF</ClientMAC>
    <ProductCodeInfo>O01101O005000\005000\100</ProductCodeInfo>
    <Input>SWIPED</Input>
</PLRequest>
```

Example Request

```
<PLRequest>
    <Command>CCSALE</Command>
    <Id>9999</Id>
    <FuelAmount>50.00</FuelAmount>
    <FuelCostPerUnit>1.00</FuelCostPerUnit>
    <PumpLaneNum>09</PumpLaneNum>
    <Odometer>0001000</Odometer>
    <Amount>2.00</Amount>
    <ClientMAC>FFFFFFFFFFFF</ClientMAC>
    <ProductCodeInfo>F01019G50000\050000\100\O01432O01000\002000\200
</ProductCodeInfo>
    <InputMethod>SWIPED</InputMethod>
</PLRequest>
```

E-commerce Transaction

Request Elements

The table below outlines the essential tags required for an eCommerce transaction.

Element Tag	Required	Type	Size	Notes
Command	Y	Fixed String	-	CCSALE
Amount	Y	Float	1-7	Amount of transaction. Two decimal places, no dollar sign – (e.g. 123.45) Positive values only.
GoodType	Y	Fixed String	1	D denotes digital goods, P denotes physical goods
CardNumber		Int	12-19	Account Number
ExpiryDate		Int	4	Expiry Date in format MMYY
AVSStreet		String	1-40	AVS Street Address for verifying card user
AVSZip		String	5-9	AVS Zip Code for verifying card user
CVV		Int	3-4	Card Verification Number for verifying card user. Called CVV2 by Visa, CVC2 by MC, and CID by Amex.

Table 23

Example Request

```
<PLRequest>
      <Command>CCSALE</Command>
      <Amount>12.00</Amount>
      <GoodType>D</GoodType>
      <CardNumber>3059990000000022</CardNumber>
      <ExpiryDate>1220</ExpiryDate>
      <CVV>111</CVV>
</PLRequest>
```

Example Response

```
<PLResponse>
        <Result>APPROVED</Result>
        <Command>CCSALE</Command>
        <ResultText>Transaction Approved</ResultText>
        <Authorization>111111</Authorization>
        <RefData>0000</RefData>
```

```
        <RecNum>2001</RecNum>
        <TerminalId>001</TerminalId>
        <MerchantId>1234567890</MerchantId>
        <ErrorCode>0000</ErrorCode>
    </PLResponse>
```

NOTE: The Industry Type in the configuration must be set to eCommerce.

Health Care Substantiation Processing

This transaction type is used for processing of expenses incurred on Flexible Spending Account (FSA) and Healthcare Reimbursement Arrangement cards (HRA). This is used exclusively for MasterCard or Visa based purchases.

As with any other credit card sale, the *<Amount>* element is used for the total transaction amount. However, in addition to the Credit Card Sale (CCSALE) elements previously discussed, the following elements may also be used to provide further detail.

Request Elements

Element Tag	Required	Type	Size	Notes
HealthCareType	N	Fixed String	-	One of either Dental, Vision, Clinical or Prescription (if Visa, not required for MasterCard transactions)
HealthCareItemAmount	N	Float	3-7	The Prescription Eligibility Amount (for MasterCard).The prescription amount (for Visa)

Table 24

Example Request

```
<PLRequest>
     <Command>CCSALE</Command>
     <Input>MANUAL</Input>
     <CardNumber>4024720012345671</CardNumber>
     <ExpiryDate>1215</ExpiryDate>
     <Amount>23.54</Amount>
     <HealthCareType>Dental</HealthCareType>
     <HealthCareItemAmount>20.00</HealthCareItemAmount>
</PLRequest>
```

Example Response

```
<PLResponse>
     <Result>APPROVED</Result>
     <Authorization>VI2000</Authorization>
     <RecNum>000124</RecNum>
     <RefData>13090500080010</RefData>
     <Language>English</Language>
     <ErrorCode>0000</ErrorCode>
     <AuthAmt>20.00</AuthAmt>
     <CardNumber>5671</CardNumber>
     <CardType>Visa</CardType>
   <CardInfo>EEC9875E133BBC8B8F781E4B10D8848BB528C0CA880802B830D78EC2EE034CBC
84E810B7754ACA16EB6D239ACF41A98855AAC646815EA49220D6FFECE7C45E87</CardInfo>
     <TransactionDate>130905</TransactionDate>
     <TransactionTime>165958</TransactionTime>
     <Command>CCSALE</Command>
```

```
      <ResultText>PARTIAL AP</ResultText>
      <MerchantId>497755528206018</MerchantId>
      <TerminalId>012</TerminalId>
      <InputMethod>MANUAL</InputMethod>
</PLResponse>
```

Gift Card Transactions

Transactions for gift cards are also known as Stored Value or Prepaid cards.

> **NOTE:** Many processors will not support the complete set of gift card transaction types. For example, the Refund and Reload transactions both result in credit being added to the card's balance. Some processors support both transactions, while others support only one of the two.

> **NOTE:** Expiry Date is no longer supported for Gift Card transactions.

Information on common element tags for requests and responses and their possible values is contained in the two (2) tables that follow. This information is not repeated for each transaction type. Only transaction-specific elements are detailed in subsequent sections.

Common Elements for Requests

The table below outlines common element tags required for all gift card request transactions.

Element Tag	Required	Type	Size	Notes
Command	Y	Fixed String	-	Specifies transaction type to be executed. Must be one of: GCACTIVATE/GCSALE/GCBALANCE/ GCVOIDACTIVATE/GCVOID/GCMINISTATEMENT/GCMASSBALANCE/GCREFUND /GCDEACTIVATE /GCCASHOUT /GCRELOAD /GCPREAUTH /GCFINAL /GCDEACTIVATE /
Id	N	String	1-20	Optional Id used by POS to track the transaction, using for example invoice# or receipt#. Sent unchanged in response, not used by POSLynx.
ClientId	N	String	1-20	The ID of the user/cashier operating the PINpad. Sent unchanged in the response, and is not used by the POSLynx.
ClientMAC	Y - if requests are sent directly to POSLynx	String	12	MAC Address of client. This id must be registered on POSLynx. This is added automatically by Precidia 'helper' applications (TNP-CG/TNP-NP), so is not always required. If requests are sent directly to POSLynx this is required.
IPAddress	N	String	7-15	Overrides the configured POSLynx IP Address. Allows controlling which POSLynx processes each transaction individually. Only applicable to TNP-CG.
IPPort	N	Numeric	4-5	Override the configured POSLynx port. Used with

				<IPAddress> to control which POSLynx and port are used to process each transaction. Only applicable to TNP-CG.
TranDetail	N	TAG	-	See section 7 - TranDetail can be used to send detailed information about the particular transaction for detailed reporting.

Table 25

Common Elements for Responses

The table below outlines common element tags required for all gift card response transactions.

Element Tag	Required	Type	Size	Notes
Result	Y	Fixed String	-	One of APPROVED / DECLINED / ERROR
Command	Y	Fixed String	-	Command returned is same as sent on the request.
ResultText	Y	String	1-20	Text to describe result – e.g. explain error or result that could be placed on POS screen.
Authorization	Y - if approved	String		Authorization code from processor.
RecNum	Y - if approved	String	4	Reference number used to refer to this transaction in future transactions. e.g. to void this transaction, RecNum should be used.
RefData	N	Int	1-20	Additional reference data returned by host. Used only be certain hosts.
ErrorCode	Y	String	4	Error code - 0000 if everything is successful. A list of error codes is yet to be defined.
Id	N	String	1-20	Optional Id used by POS to track the transaction e.g. invoice # or receipt #. Sent unchanged in response. If sent in request it will be returned in response.
ClientId	N	String	1-20	The ID of the user/cashier operating the PINpad. Sent unchanged in the response, and is not used by the POSLynx.
CardNumber	Y	String	4	Returns last four digits of card number – for display if needed.
CardType	Y	Fixed String	-	Type of card sent in request. Will display 'Gift' for gift cards. Based on card ranges defined in POSLynx configuration.
MerchantId	N	String	1-20	Merchant Id used to process transaction with host
TerminalId	N	String	1-20	Terminal Id used to process transaction with host
Receipt / Receipt#	N			If enabled in POSLynx, a printable host-dependent Receipt will be part of the response. Not shown in examples, see *Chapter 6: Receipts*.
TransactionDate	N	Int	6	Date of Transaction – MMDDYY format. Not sent by all host configurations.

TransactionTime	N	Int	6	Time of Transaction – HHMMSS format. Not sent by all host configurations.

Table 26

Gift Card Activation

A gift card must be activated before use. The activation registers the card with the host and places a credit value on the card. This credit can then be later used to purchase items from the merchant. The following table outlines the associated element tags.

Element Tag	Required	Type	Size	Notes
Command	Y	Fixed String	-	GCACTIVATE
Amount	Y	Float	1-7	Amount to be refunded. Two decimal places, no dollar sign – (e.g. 123.45) Positive values only.
Input	Y	Fixed String	-	Specifies type of card data given. Must be one of : SWIPED/MANUAL/EXTERNAL
Track2	Y - If Swiped	String	1-40	Swiped track2 data - without sentinel characters. Required if SWIPED.
Track1	N	String	1-80	Swiped track1 data - without start/end sentinel characters.
CardNumber	Y - If Manual	Int	12-19	Account number. Required if MANUAL.
Track2EndRange	Y – If multiple cards are to be activated	String	1-40	Used with Gift Card request messages to allow multiple gift card transactions to be processed in a single request. Denotes the track data of the last card in the range.

Table 27

Example Request

A swiped transaction:

```
<PLRequest>
    <Command>GCACTIVATE</Command>
    <Amount>20.00</Amount>
    <Id>9999</Id>
    <Input>SWIPED</Input>
    <Track2>6010222233332244=00010004070779134</Track2>
    <Track1>6010222233332244^^00010004070779134</Track1>
</PLRequest>
```

A manually keyed transaction:

```
<PLRequest>
    <Command>GCACTIVATE</Command>
    <Amount>20.00</Amount>
    <Id>9999</Id>
```

```
    <Input>MANUAL</Input>
    <CardNumber>6010222233332244</CardNumber>
 </PLRequest>
```

Example Response

An approved transaction response:

```
<PLResponse>
    <Result>APPROVED</Result>
    <Command>GCACTIVATE</Command>
    <Id>9999</Id>
    <ResultText>Transaction Approved</ResultText>
    <Authorization>123456</Authorization>
    <RecNum>1001</RecNum>
    <ErrorCode>0000</ErrorCode>
    <Balance>20.00</Balance>
    <RefData>0000</RefData>
    <CardNumber>4444</CardNumber>
    <CardType>Gift</CardType>
    <MerchantId>1234567890</MerchantId>
    <TerminalId>012</TerminalId>
</PLResponse>
```

A declined transaction response:

```
<PLResponse>
    <Result>DECLINED</Result>
    <Command> GCACTIVATE </Command>
    <Id>9999</Id>
    <ResultText>Internal Error</ResultText>
    <ErrorCode>0100</ErrorCode>
</PLResponse>
```

Gift Card Sale

This transaction type processes a payment on a gift card. Can be swiped or manually entered. The transaction is finalized on the customer card and funds are transferred to the merchant after the batch is closed.

The command for this transaction is GCSALE.

GCSALE

This command is used in transactions where the balance on the card is greater than (or equal to) the transaction amount. Should the card balance be less than the transaction amount, the transaction will be declined.

Element Tag	Required	Type	Size	Notes
Command	Y	Fixed String	-	GCSALE
Amount	Y	Float	1-7	Amount of transaction. Two decimal places, no dollar sign – (e.g. 123.45) Positive values only.
Input	Y	Fixed String	-	Specifies type of card data given. Must be one of : SWIPED/MANUAL/EXTERNAL
Track2	Y - If Swiped	String	1-40	Swiped track2 data - without sentinel characters. Required if SWIPED
Track1	N	String	1-80	Swiped track1 data - without start/end sentinel characters.
CardNumber	Y - If Manual	Int	12-19	Account number. Required if MANUAL.

Table 28

Example Request

A swiped transaction:

```
<PLRequest>
    <Command>GCSALE</Command>
    <Amount>10.00</Amount>
    <Id>9999</Id>
    <Input>SWIPED</Input>
    <Track2>6010222233332244=00010004007077913 4</Track2>
    <Track1>6010222233332244^^00010004007077913 4</Track1>
</PLRequest>
```

A manually keyed transaction:

```
<PLRequest>
```

```
    <Command>GCSALE</Command>
    <Amount>10.00</Amount>
    <Id>9999</Id>
    <Input>MANUAL</Input>
    <CardNumber>6010222233332244</CardNumber>
 </PLRequest>
```

Example Response

An approved transaction response:

```
<PLResponse>
    <Result>APPROVED</Result>
    <Command>GCSALE</Command>
    <ResultText>Transaction Approved</ResultText>
    <Id>9999</Id>
    <AuthAmt>10.00</AuthAmt>
    <Authorization>123456</Authorization>
    <RefData>0000</RefData>
    <RecNum>1001</RecNum>
    <ErrorCode>0000</ErrorCode>
    <Balance>16.00</Balance>
    <CardNumber>4444</CardNumber>
    <CardType>Gift</CardType>
    <MerchantId>1234567890</MerchantId>
    <TerminalId>012</TerminalId>
</PLResponse>
```

A declined transaction response:

```
<PLResponse>
    <Result>DECLINED</Result>
    <Command>CCSALE </Command>
    <Id>9999</Id>
    <ResultText>Insufficient Funds</ResultText>
    <ErrorCode>0100</ErrorCode>
</PLResponse>
```

Gift Card Balance/ Mini Statement/ Mass Balance

This transaction determines the cash available on the gift card. Can be swiped or manually entered. The 'mini-statement' request will provide details on recent gift card transactions (not supported by all processors).

The 'mass balance' request provides the balance of a series of gift cards (not supported by all processors). The following table outlines the associated tags.

Element Tag	Required	Type	Size	Notes
Command	Y	Fixed String	-	GCBALANCE or GCMASSBALANCE or GCMINISTATEMENT
Input	Y	Fixed String	-	Specifies type of card data given. Must be one of : SWIPED/MANUAL/EXTERNAL
Track2	Y - If Swiped	String	1-40	Swiped track2 data - without sentinel characters. Required if SWIPED. If GCMASSBALANCE, will be the starting range
Track1	N	String	1-80	Swiped track1 data - without start/end sentinel characters.
CardNumber	Y - If Manual	Int	12-19	Account number. Required if MANUAL.
Track2EndRange	Y - If GCMASSBALANCE	String	1-40	Used with Gift Card request messages to allow multiple gift card transactions to be processed in a single request. Denotes the track data of the last card in the range.

Table 29

Example Request

A swiped balance transaction:

```
<PLRequest>
    <Command>GCBALANCE</Command>
    <Id>9999</Id>
    <Input>SWIPED</Input>
    <Track2>6010222233332244=00010004007077913 4</Track2>
    <Track1>6010222233332244^^00010004007077913 4</Track1>
</PLRequest>
```

A manually keyed balance transaction:

```
<PLRequest>
    <Command>GCBALANCE</Command>
    <Input>MANUAL</Input>
    <Id>9999</Id>
    <CardNumber>6010222233332244</CardNumber>
</PLRequest>
```

A swiped mini-transaction:

```
<PLRequest>
      <Command>GCMINISTATEMENT</Command>
      <Id>9999</Id>
      <Track2>6272110090230009=4912101000002170</Track2>
      <Input>SWIPED</Input>
</PLRequest>
```

Example Response

An approved balance transaction response:

```
<PLResponse>
    <Result>APPROVED</Result>
    <Command>GCBALANCE</Command>
    <Id>9999</Id>
    <ResultText>Transaction Approved</ResultText>
    <Authorization>123456</Authorization>
    <RefData>0000</RefData>
    <ErrorCode>0000</ErrorCode>
    <Balance>30.00</Balance>
    <CardNumber>4444</CardNumber>
    <CardType>Gift</CardType>
    <MerchantId>1234567890</MerchantId>
    <TerminalId>012</TerminalId>
</PLResponse>
```

A declined balance transaction response:

```
<PLResponse>
    <Result>DECLINED</Result>
    <Command>GCBALANCE</Command>
    <Id>9999</Id>
    <ResultText>Inactive Acct</ResultText>
    <ErrorCode>0100</ErrorCode>
</PLResponse>
```

Gift Card Refund

This transaction will process a refund/return on a gift card, which will add cash value to the card.

Element Tag	Required	Type	Size	Notes
Command	Y	Fixed String	-	GCREFUND
Amount	Y	Float	1-7	Amount to add to the gift card. Two decimal places, no dollar sign – (e.g. 123.45) Positive values only.
Input	Y	Fixed String	-	Specifies type of card data given. Must be one of : SWIPED/MANUAL/EXTERNAL
Track2	Y - If Swiped	String	1-40	Swiped track2 data - without sentinel characters. Required if SWIPED
Track1	N	String	1-80	Swiped track1 data - without start/end sentinel characters.
RefNum	N	Int	6	Reference number of original transaction. This is normally not needed, but for some processors is required (ie: Ernex).

Table 30

Example Request

A swiped transaction:

```
<PLRequest>
    <Command>GCREFUND</Command>
    <Input>SWIPED</Input>
    <Id>9999</Id>
    <Track2>6010222233332244=00010004007077913 4</Track2>
    <Track1>6010222233332244^^00010004007077913 4</Track1>
    <Amount>25.00</Amount>
    <RefNum>123456</RefNum>
</PLRequest>
```

A manually keyed transaction:

```
<PLRequest>
    <Command>GCREFUND</Command>
    <Input>MANUAL</Input>
    <Id>9999</Id>
    <CardNumber>6010222233332244</CardNumber>
    <Amount>5.00</Amount>
```

```
</PLRequest>
```

Example Response

An approved transaction response:

```
<PLResponse>
    <Result>APPROVED</Result>
    <Command>GCREFUND</Command>
    <Id>9999</Id>
    <ResultText>Transaction Approved</ResultText>
    <Authorization>123456</Authorization>
    <RefData>0000</RefData>
    <ErrorCode>0000</ErrorCode>
    <Balance>50.00</Balance>
    <CardNumber>4444</CardNumber>
    <CardType>Gift</CardType>
    <MerchantId>1234567890</MerchantId>
    <TerminalId>012</TerminalId>
</PLResponse>
```

A declined transaction response:

```
<PLResponse>
    <Result>DECLINED</Result>
    <Command>GCREFUND</Command>
    <Id>9999</Id>
    <ResultText>Inactive Acct</ResultText>
    <ErrorCode>0100</ErrorCode>
</PLResponse>
```

Gift-Card Reload

This transaction adds cash value to the gift card. The table below outlines the associated element tags.

Element Tag	Required	Type	Size	Notes
Command	Y	Fixed String	-	GCRELOAD
Input	Y	Fixed String	-	Specifies type of card data given. Must be one of : SWIPED/MANUAL/EXTERNAL
Track2	Y - If Swiped	String	1-40	Swiped track2 data - without sentinel characters. Required if SWIPED.
Track1	N	String	1-80	Swiped track1 data - without start/end sentinel characters.
CardNumber	Y - If Manual	Int	12-19	Account number. Required if MANUAL.
Track2EndRange	Y – If multiple cards are to be reloaded	String	1-40	Used with Gift Card request messages to allow multiple gift card transactions to be processed in a single request. Denotes the track data of the last card in the range.
Amount	Y	Float	1-7	Amount to add to the gift card. Two decimal places, no dollar sign – (e.g. 123.45) Positive values only.

Table 31

Example Request

A swiped transaction:

```
<PLRequest>
    <Command>GCRELOAD</Command>
    <Id>9999</Id>
    <Input>SWIPED</Input>
    <Track2>6010222233332244=00010004007077913 4</Track2>
    <Track1>6010222233332244^^00010004007077913 4</Track1>
    <Amount>25.00</Amount>
</PLRequest>
```

A manually keyed transaction:

```
<PLRequest>
    <Command>GCRELOAD</Command>
    <Input>MANUAL</Input>
    <Id>9999</Id>
    <CardNumber>6010222233332244</CardNumber>
```

```
      <Amount>5.00</Amount>
  </PLRequest>
```

Example Response

An approved transaction response:

```
<PLResponse>
    <Result>APPROVED</Result>
    <Command>GCRELOAD</Command>
    <Id>9999</Id>
    <ResultText>Transaction Approved</ResultText>
    <Authorization>123456</Authorization>
    <RefData>0000</RefData>
    <ErrorCode>0000</ErrorCode>
    <Balance>50.00</Balance>
    <CardNumber>4444</CardNumber>
    <CardType>Gift</CardType>
    <MerchantId>1234567890</MerchantId>
    <TerminalId>012</TerminalId>
</PLResponse>
```

A declined transaction response:

```
<PLResponse>
    <Result>DECLINED</Result>
    <Command>GCRELOAD</Command>
    <Id>9999</Id>
    <ResultText>Inactive Acct</ResultText>
    <ErrorCode>0100</ErrorCode>
</PLResponse>
```

Gift Card Void Activate

This transaction voids a completed gift card activation. For example, in cases where a payment for the gift card cannot be processed, the gift card must be deactivated.

Element Tag	Required	Type	Size	Notes
Command	Y	Fixed String	-	GCVOIDACTIVATE
Amount	Y	Float	1-7	Amount of previous activate transaction. Two decimal places, no dollar sign – (e.g. 123.45) Positive values only.
RecNum	Y	String	4	Reference to the original transaction to be voided.

Table 32

Example Request

```
<PLRequest>
    <Command>GCVOIDACTIVATE</Command>
    <Id>9999</Id>
    <Amount>20.00</Amount>
    <RecNum>1004</RecNum>
</PLRequest>
```

Example Response

An approved transaction response:

```
<PLResponse>
    <Result>APPROVED</Result>
    <Command>GCVOIDACTIVATE</Command>
    <Id>9999</Id>
    <ResultText>Transaction Approved</ResultText>
    <Authorization>123456</Authorization>
    <RefData>0000</RefData>
    <ErrorCode>0</ErrorCode>
    <CardNumber>4444</CardNumber>
    <CardType>Gift</CardType>
    <MerchantId>1234567890</MerchantId>
    <TerminalId>012</TerminalId>
</PLResponse>
```

A declined transaction response:

```
<PLResponse>
    <Result>DECLINED</Result>
    <Command>GCVOIDACTIVATE</Command>
    <Id>9999</Id>
    <ResultText>Inactive Acct</ResultText>
    <ErrorCode>100</ErrorCode>
</PLResponse>
```

Gift Card Void

This transaction voids a completed Gift Card Sale, Refund, or Reload transaction. The table below outlines the associated element tags.

Element Tag	Required	Type	Size	Notes
Command	Y	Fixed String	-	GCVOID
Amount	Y	Float	1-7	Amount of previous sale transaction. Two decimal places, no dollar sign – (e.g. 123.45) Positive values only.
RecNum	Y	String	4	Reference to the original transaction to be voided.
EndRange	Y – If void multiple cards are to be voided	String	1-40	The ending range

Table 33

Example Request

```
<PLRequest>
    <Command>GCVOID</Command>
    <Amount>5.00</Amount>
    <Id>9999</Id>
    <RecNum>1006</RecNum>
</PLRequest>
```

Element Tag	Required	Type	Size	Notes
Balance	Y	Float	3-7	Cash Balance remaining on the card.

Table 34

Example Response

An approved transaction response:

```
<PLResponse>
    <Result>APPROVED</Result>
    <Command>GCVOID</Command>
    <Id>9999</Id>
    <ResultText>Transaction Approved</ResultText>
    <Authorization>123456</Authorization>
    <RefData>0000</RefData>
    <ErrorCode>0</ErrorCode>
```

```
    <Balance>25.00</Balance>
    <CardNumber>4444</CardNumber>
    <CardType>Gift</CardType>
    <MerchantId>1234567890</MerchantId>
    <TerminalId>012</TerminalId>
</PLResponse>
```

A declined transaction response:

```
<PLResponse>
    <Result>DECLINED</Result>
    <Command>GCVOID</Command>
    <Id>9999</Id>
    <ResultText>Inactive Acct</ResultText>
    <ErrorCode>100</ErrorCode>
</PLResponse>
```

Gift Card Cashout

This transaction clears the balance on the gift card.

The table below outlines the associated element tags.

Element Tag	Required	Type	Size	Notes
Command	Y	Fixed String	-	GCCASHOUT
Input	Y	Fixed String	-	Specifies type of card data given. Must be one of : SWIPED/MANUAL/EXTERNAL
Track2	Y - If Swiped	String	1-40	Swiped track2 data - without sentinel characters. Required if SWIPED.
Track1	N	String	1-80	Swiped track1 data - without start/end sentinel characters.
CardNumber	Y - If Manual	Int	12-19	Account number. Required if MANUAL.
Track2EndRange	Y – If multiple cards are being Cashed out	String	1-40	Used with Gift Card request messages to allow multiple gift card transactions to be processed in a single request. Denotes the track data of the last card in the range.

Table 35

Example Request

A swiped transaction:

```
<PLRequest>
    <Command>GCCASHOUT</Command>
    <Id>9999</Id>
    <Input>SWIPED</Input>
    <Track2>6010222233332244=00010004007077913 4</Track2>
    <Track1>6010222233332244^^00010004007077913 4</Track1>
</PLRequest>
```

A manually keyed transaction:

```
<PLRequest>
    <Command>GCCASHOUT</Command>
    <Id>9999</Id>
    <Input>MANUAL</Input>
    <CardNumber>6010222233332244</CardNumber>
</PLRequest>
```

Element Tag	Required	Type	Size	Notes
Balance	Y	Float	3-7	Cash Balance remaining on the card. (should be 0.00)
PrevBalance	Y	Float	3-7	Cash Balance that was now removed from the card.

Table 36

Example Response

An approved transaction response:

```
<PLResponse>
    <Result>APPROVED</Result>
    <Command>GCCASHOUT</Command>
    <Id>9999</Id>
    <ResultText>Transaction Approved</ResultText>
    <Authorization>123456</Authorization>
    <RecNum>1011</RecNum>
    <ErrorCode>0000</ErrorCode>
    <Balance>0.00</Balance>
    <PrevBalance>12.30</PrevBalance>
    <RefData>12345678</RefData>
    <CardNumber>4444</CardNumber>
    <CardType>Gift</CardType>
```

```
    <MerchantId>1234567890</MerchantId>
    <TerminalId>012</TerminalId>
</PLResponse>
```

Gift Card Deactivate

This transaction resets the card to a deactivated state, clears the balance and returns any balance to the card at the time of deactivation. The card must be activated again before use. In some cases, a reset must be performed prior to gift card activation.

> **NOTE:** Not supported by all processors.

The table below outlines the associated element tags.

Element Tag	Required	Type	Size	Notes
Command	Y	Fixed String	-	GCDEACTIVATE
Input	Y	Fixed String	-	Specifies type of card data given. Must be one of : SWIPED/MANUAL/EXTERNAL
Track2	Y - If Swiped	String	1-40	Swiped track2 data - without sentinel characters. Required if SWIPED.
Track1	N	String	1-80	Swiped track1 data - without start/end sentinel characters.
CardNumber	Y - If Manual	Int	12-19	Account number. Required if MANUAL.
Track2EndRange	Y – If multiple cards are being deactivated	String	1-40	Used with Gift Card request messages to allow multiple gift card transactions to be processed in a single request. Denotes the track data of the last card in the range.

Table 37

Example Request

A swiped transaction:

```
<PLRequest>
    <Command>GCDEACTIVATE</Command>
    <Input>SWIPED</Input>
    <Id>9999</Id>
    <Track2>6010222233332244=00010004007077913 4</Track2>
    <Track1>6010222233332244^^00010004007077913 4</Track1>
</PLRequest>
```

A manually keyed transaction:

```
<PLRequest>
    <Command>GCDEACTIVATE</Command>
    <Input>MANUAL</Input>
```

```
    <Id>9999</Id>
    <CardNumber>6010222233334444</CardNumber>
</PLRequest>
```

Element Tag	Required	Type	Size	Notes
Balance	Y	Float	3-7	Cash Balance remaining on the card. (should be 0.00)
PrevBalance	Y	Float	3-7	Cash Balance that was now removed from the card.

Table 38

Example Response

An approved transaction response:

```
<PLResponse>
    <Result>APPROVED</Result>
    <Command>GCCASHOUT</Command>
    <Id>9999</Id>
    <ResultText>Transaction Approved</ResultText>
    <Authorization>123456</Authorization>
    <RecNum>1011</RecNum>
    <ErrorCode>0000</ErrorCode>
    <Balance>0.00</Balance>
    <PrevBalance>12.30</PrevBalance>
    <RefData>12345678</RefData>
    <CardNumber>4444</CardNumber>
    <CardType>Gift</CardType>
    <MerchantId>1234567890</MerchantId>
    <TerminalId>012</TerminalId>
</PLResponse>
```

Gift Card Pre-Authorize Transaction

This transaction is used to authorize a transaction without actually placing a charge on the gift card. It ensures there is credit available for the transaction when finalized, putting a temporary hold on the gift card funds for the user. Can be swiped or manually keyed.

> **NOTE:** Not supported by all processors.

Element Tag	Required	Type	Size	Notes
Command	Y	Fixed String	-	GCPREAUTH
Input	Y	Fixed String	-	Specifies type of card data given. Must be one of : SWIPED/MANUAL/EXTERNAL
AuthAmt	Y	Float	3-7	Amount to be authorized for this transaction. POS may include an estimated gratuity in this amount to ensure that finalized transaction will be approved. Two decimal places, no dollar sign – (e.g. 123.45) Positive values only.
Track2	Y - If Swiped	String	1-40	Swiped track2 data - without sentinel characters. Required if SWIPED
Track1	N	String	1-80	Swiped track1 data - without start/end sentinel characters.
CardNumber	Y - If Manual	Int	12-19	Account number. Required if MANUAL.

Table 39

Example Request

A swiped transaction:

```
<PLRequest>
      <Command>GCPREAUTH</Command>
      <Input>SWIPED</Input>
      <AuthAmt>0.65</AuthAmt>
      <Track2>5858837401000004=39121010123456789012</Track2>
      <Id>9999</Id>
</PLRequest>
```

A manually keyed transaction:

```
<PLRequest>
      <Command>GCPREAUTH</Command>
      <Input>MANUAL</Input>
      <AuthAmt>0.65</AuthAmt>
```

```
        <CardNumber>5858837401000004</CardNumber>
        <Id>9999</Id>
</PLRequest>
```

Gift Card Pre-Auth Completion Transaction

This transaction finalizes the prior approved transaction. The funds will be placed on the customer's gift card and when the batch is closed, the funds will be transferred to the merchant. If this is not done, the Pre-Auth transaction will eventually timeout and nothing will take place.

This transaction is tied to the previous transaction via the *<RecNum>* tag. The following table outlines the associated element tags.

Element Tag	Required	Type	Size	Notes
Command	Y	Fixed String	-	GCFINAL
Amount	Y	Float	1-7	Actual Purchase Amount of this transaction. Two decimal places, no dollar sign – (e.g. 123.45).Positive values only.
Gratuity	N	Float	3-7	Amount of gratuity for this transaction. Total transaction amount will be for total of Amount+Gratuity. Positive values only.
AuthAmt	Y	Float	3-7	Full Amount of this transaction (Amount+Gratuity). Two decimal places, no dollar sign – (e.g. 123.45). Positive values only.
RecNum	Y	String	4	Reference to the original Pre-Auth transaction.

Table 40

Example Request

A swiped transaction:

```
<PLRequest>
        <Command>GCFINAL</Command>
        <Amount>0.55</Amount>
        <Track2>5858836401000004=49121010123456789012</Track2>
        <Input>SWIPED</Input>
        <RecNum>3421</RecNum>
        <Id>9999</Id>
</PLRequest>
```

A manually keyed transaction:

```
<PLRequest>
        <Command>GCFINAL</Command>
        <Input>MANUAL</Input>
```

```
      <Amount>0.55</Amount>
      <CardNumber>5858836401000004</CardNumber>
      <RecNum>3421</RecNum>
      <Id>9999</Id>
</PLRequest>
```

Debit Transactions

Debit Transactions are supported, with the POSLynx controlling the PIN Pad. The Verifone 1000SE and SC5000 PIN Pads are currently supported. The PIN Pad can be connected to the POSLynx serial port or to the serial port of the POS system. Multiple PIN Pad devices can be controlled by connecting the PIN Pads to the network through the Precidia *iPocket* device.

When a debit request is made, the POSLynx will control the PIN Pad and, to confirm the transaction with the host processor, prompt the user for a PIN. If the PIN Pad is connected to the POS system, then one of the Precidia helper applications (TNP-CG, TNP-NP) must be used to control the PIN Pad.

Specific information on each transaction type is found in the following sections. Information on common element tags for requests and responses and their possible values, is contained in the two (2) tables that follow. This information is not repeated for each transaction type. Only the transaction-specific elements are discussed for each type.

Common Elements for Requests

Tags common to all debit transaction requests, are outlined in the table below.

Element Tag	Required	Type	Size	Notes
Command	Y	Fixed String	-	Specifies transaction type to be executed. Must be one of: DCSALE/DCREFUND
Id	N	String	1-20	Optional Id used by POS to track the transaction e.g. invoice # or receipt #. Sent unchanged in response, it is not used by POSLynx.
ClientId	N	String	1-20	The ID of the user/cashier operating the PINpad. Sent unchanged in the response, and is not used by the POSLynx.
ClientMAC	Y - if requests are sent directly to POSLynx.	String	12	MAC Address of client. This id must be registered on POSLynx. This is added automatically by Precidia 'helper' applications (TNP-CG/TNP-NP), so is not always required. If requests are sent directly to POSLynx this is required.
Amount	Y	Float	1-7	Amount of transaction. Two decimal places, no dollar sign – (e.g. 123.45). Positive values only.
Input	Y	Fixed String	-	Specifies type of card data given. Must be one of : SWIPED/MANUAL/EXTERNAL
Track2	Y - If Swiped	String	1-40	Swiped track2 data - without sentinel characters. Required if SWIPED.
Track1	N	String	1-80	Swiped track1 data - without start/end sentinel characters.
CardNumber	Y - If Manual	Int	12-19	Account number. Required if MANUAL.
ExpiryDate	N	Int	4	Expiry Date in format MMYY. Not always

				required for Gift card transactions.

Table 41

Common Elements for Responses

Tags common to all debit transaction responses, are outlined in the table below.

Element Tag	Required	Type	Size	Notes
Result	Y	Fixed String	-	One of APPROVED / DECLINED / ERROR
Command	Y	Fixed String	-	Command returned is same as sent on the request.
ResultText	Y	String	1-20	Text to describe result – e.g. explain error or result that could be placed on POS screen.
Authorization	Y - if approved	String	1-20	Authorization code from processor.
AuthAmt	Y	Float	3-7	Amount processed for the transaction.
RecNum	Y - if approved	String	4	Reference number used to refer to this transaction in future transactions. e.g. to void this transaction, RecNum should be used.
RefData	N	Int	1-20	Additional reference data returned by host – may not be used – or may only be used by certain hosts.
ErrorCode	Y	String	4	Error code - 0000 if everything is successful.A list of error codes is yet to be defined.
Id	N	String	1-20	Optional Id used by POS to track the transaction e.g. invoice # or receipt #. Sent unchanged in response. If sent in request it will be returned in response.
ClientId	N	String	1-20	The ID of the user/cashier operating the PINpad. Sent unchanged in the response, and is not used by the POSLynx.
CardNumber	Y	String	4	Returns last four digits of card number – for display if needed.
CardType	Y	Fixed String	-	Type of card sent in request. Will display 'OtherCard' for debit. Based on card ranges defined in POSLynx configuration.
MerchantId	N	String	1-20	Merchant Id used to process transaction with host
TerminalId	N	String	1-20	Terminal Id used to process transaction with host
Receipt / Receipt#	N			If enabled in POSLynx, a printable host dependant Receipt will be part of the response.Not shown in examples, see section 6.
TransactionDate	N	Int	6	Date of Transaction – MMDDYY format. Not sent by all host configurations.
TransactionTime	N	Int	6	Time of Transaction – HHMMSS format. Not sent by all host configurations.
InputMethod	N	String	1-20	How the card was entered. One of SWIPED/MANUAL/TOKEN/CHIP/TAP/RFID. See

				'Card Entry Options' in Chapter 5 for more detail.

Table 42

Debit Sale

This transaction processes a debit sale or payment. This effectively removes the transaction amount from the customer account, and will increase the batch amount for the merchant. The card holder will be prompted for a PIN number on the PIN Pad.

Request Elements

Please note that this list is not exhaustive. Other possible elements are covered in 'Common Elements' for Debit Card Requests.

Element Tag	Required	Type	Size	Notes
Command	Y	Fixed String	-	DCSALE
Cashback	N	Float	3-7	Amount of extra cash to be added to total amount of transaction.
Input	Y	String	1-20	How the card was entered. One of SWIPED/MANUAL/EXTERNAL/TOKEN. See 'Card Entry Options' in Chapter 5 for more detail.
Id	N	String	1-20	Optional Id used by POS to track the transaction e.g. invoice # or receipt #. Sent unchanged in response. If sent in request it will be returned in response.

Table 43

Example Request

A swiped transaction:

```
<PLRequest>
    <Command>DCSALE</Command>
    <Amount>1.00</Amount>
    <Input>SWIPED</Input>
    <Track2>9988999800002773=1012101543211234</Track2>
    <Track1>9988999800002773^LAST/FIRST^10121015432112345678</Track1>
    <Id>9999</Id>
</PLRequest>
```

Response Elements

Please note that this list is not exhaustive. Other possible elements are covered in 'Common Elements' for Debit Card Responses.

Element Tag	Required	Type	Size	Notes
AuthAmt	Y	Float	3-7	Amount processed and removed from balance (equal to Amount + CashBack)
Amount	Y	Float	1-7	Amount of actual Sale
Cashback	N	Float	3-7	Amount of Cash given back to customer.
CustomerName	N	String	1-20	Customer name found in Track1 input. Only returned if Track1 data sent in request.
InputMethod	N	String	1-20	How the card was entered. One of SWIPED/MANUAL/TOKEN/CHIP/TAP/RFID. See 'Card Entry Options' in Chapter 5 for more detail.

Table 44

Example Response

An approved transaction response:

```
<PLResponse>
    <Result>APPROVED</Result>
    <CardNumber>2773</CardNumber>
    <CardType>OtherCard</CardType>
    <Authorization>094449</Authorization>
    <Amount>10.00</Amount>
    <Cashback>5.00</Cashback>
    <AuthAmt>15.00</AuthAmt>
    <RecNum>2002</RecNum>
    <RefData>00000002</RefData>
    <Command>DCSALE</Command>
    <ResultText>Transaction Approved</ResultText>
    <ErrorCode>0000</ErrorCode>
    <MerchantId>1234567890</MerchantId>
    <TerminalId>012</TerminalId>
    <Id>9999</Id>
    <InputMethod>SWIPED</InputMethod>
</PLResponse>
```

A failed transaction response:

```
<PLResponse>
    <Result>ERROR</Result>
    <CardNumber>2773</CardNumber>
    <CardType>OtherCard</CardType>
    <Command>DCSALE</Command>
    <ResultText>FAIL READ PIN</ResultText>
```

```
    <ErrorCode>0126</ErrorCode>
    <Id>9999</Id>
    <InputMethod>SWIPED</InputMethod>
</PLResponse>
```

Debit Refund

This transaction processes a debit refund or return. This effectively adds the transaction amount from the customer account, and will decrease the batch amount for the merchant. The cardholder will be prompted for a PIN number on the PIN Pad.

Request Elements

Please note that this list is not exhaustive. Other possible elements are covered in 'Common Elements' for Debit Card Requests.

Element Tag	Required	Type	Size	Notes
Command	Y	Fixed String	-	DCREFUND
Amount	Y	Float	1-7	Amount of transaction. Two decimal places, no dollar sign – (e.g. 123.45). Positive values only.
Input	Y	String	1-20	How the card was entered. One of SWIPED/MANUAL/EXTERNAL/TOKEN. See 'Card Entry Options' in Chapter 5 for more detail.
Id	N	String	1-20	

Table 45

Example Request

A swiped transaction:

```
<PLRequest>
    <Command>DCREFUND</Command>
    <Amount>1.00</Amount>
    <Input>SWIPED</Input>
    <Track2>9988999800002773=1012101543211234</Track2>
    <Track1>9988999800002773^LAST/FIRST^10121015432112345678</Track1>
    <Id>9999</Id>
</PLRequest>
```

A manually keyed transaction:

```
<PLRequest>
    <Command>DCREFUND</Command>
    <Amount>1.00</Amount>
    <Input>MANUAL</Input>
    <CardNumber>9988999800002773</CardNumber>
    <ExpiryDate>1212</ExpiryDate>
```

```
    <Id>9999</Id>
</PLRequest>
```

Response Elements

Please note that this list is not exhaustive. Other possible elements are covered in 'Common Elements' for Debit Card Responses.

Element Tag	Required	Type	Size	Notes
CustomerName	N	String	1-20	Customer name found in Track1 input. Only returned if Track1 data sent in request.
Id	N	String	1-20	Optional Id used by POS to track the transaction e.g. invoice # or receipt #. Sent unchanged in response. If sent in request it will be returned in response.
InputMethod	N	String	1-20	How the card was entered. One of SWIPED/MANUAL/TOKEN/CHIP/TAP/RFID. See 'Card Entry Options' in Chapter 5 for more detail.

Table 46

Example Response

An approved transaction response:

```
<PLResponse>
    <Result>APPROVED</Result>
    <CardNumber>2773</CardNumber>
    <CardType>OtherCard</CardType>
    <Authorization>095426</Authorization>
    <AuthAmt>1.00</AuthAmt>
    <RecNum>2004</RecNum>
    <RefData>00000033</RefData>
    <Command>DCREFUND</Command>
    <ResultText>Transaction Approved</ResultText>
    <ErrorCode>0000</ErrorCode>
    <MerchantId>1234567890</MerchantId>
    <TerminalId>012</TerminalId>
    <Id>9999</Id>
    <InputMethod>SWIPED</InputMethod>
</PLResponse>
```

A failed transaction response:

```
<PLResponse>
    <Result>ERROR</Result>
```

```
    <CardNumber>2773</CardNumber>
    <CardType>OtherCard</CardType>
    <Command>DCREFUND</Command>
    <ResultText>FAIL READ PIN</ResultText>
    <ErrorCode>0126</ErrorCode>
    <Id>9999</Id>
    <InputMethod>SWIPED</InputMethod>
</PLResponse>
```

PIN-Less Debit

This command is currently only supported by the Global Payments East Host when used in conjunction with the Equinox PIN Pad.

A manually entered debit transaction is deemed to be a PIN-Less debit transaction, where no PIN entry is required.

Request Elements

Please note that this list is not exhaustive. Other possible elements are covered in 'Common Elements' for Debit Card Requests.

Element Tag	Required	Type	Size	Notes
Command	Y	Fixed String	-	DCSALE
Amount	Y	Float	1-7	Amount of transaction. Two decimal places, no dollar sign – (e.g. 123.45). Positive values only.
Input	Y	String	1-20	How the card was entered. In the case of PIN-less debit, this value will always be 'MANUAL'.
CardNumber	Y - If Manual	Int	12-19	Account number. Required if MANUAL.
ExpiryDate	N	Int	4	Expiry Date in format MMYY. Not always required for Gift card transactions.
Id	N	String	1-20	Optional Id used by POS to track the transaction e.g. invoice # or receipt #. Sent unchanged in response, it is not used by POSLynx.

Table 47

Example Request

A sample PIN-Less transaction request:

```
<PLRequest>
    <Command>DCSALE</Command>
    <Input>MANUAL</Input>
    <Amount>4.93</Amount>
    <CardNumber>4011190070070071</CardNumber2>
    <ExpiryDate>1512</ExpiryDate>
    <Id>9999</Id>
</PLRequest>
```

Response Elements

Please note that this list is not exhaustive. Other possible elements are covered in 'Common Elements' for Debit Card Responses.

Element Tag	Required	Type	Size	Notes
Id	N	String	1-20	Optional Id used by POS to track the transaction e.g. invoice # or receipt #. Sent unchanged in response. If sent in request it will be returned in response.
InputMethod	N	String	1-20	How the card was entered. In the case of PIN-less debit, this will always be 'MANUAL'.

Table 48

Example Response

An approved transaction response:

```
<PLResponse>
    <Result>APPROVED</Result>
    <CardNumber>2773</CardNumber>
    <CardType>OtherCard</CardType>
    <Authorization>094449</Authorization>
    <Amount>10.00</Amount>
    <Cashback>5.00</Cashback>
    <AuthAmt>15.00</AuthAmt>
    <RecNum>2002</RecNum>
    <RefData>00000002</RefData>
    <Command>DCSALE</Command>
    <ResultText>Transaction Approved</ResultText>
    <ErrorCode>0000</ErrorCode>
    <MerchantId>1234567890</MerchantId>
    <TerminalId>012</TerminalId>
    <Id>9999</Id>
    <InputMethod>MANUAL</InputMethod>
</PLResponse>
```

Cash Transactions

Cash Transactions differ from other transaction types. There is no host approval required and very few error conditions can occur. A cash transaction is usually sent to the POSLynx to track sales. The sales are sent as statistics to the NetVu server and will be available in MerchantVu reports. The reports subsequently become of greater value to businesses with a high percentage of cash sales.

Specific information on each transaction type is found in the following sections. Information on common element tags for requests and responses and their possible values is contained in the two (2) tables that follow. This information is not repeated for each transaction type. Only transaction-specific elements are discussed for each type.

Common Elements for Requests

Tags common to all cash transaction requests, are outlined in the table below.

Element Tag	Req	Type	Size	Notes
Command	Y	Fixed String	-	Specifies transaction type to be executed. Must be one of: CASHSALE/CASHREFUND
Id	N	String	1-20	Optional Id used by POS to track the transaction e.g. invoice # or receipt #. Sent unchanged in response, it is not used by POSLynx.
ClientId	N	String	1-20	The ID of the user/cashier operating the PINpad. Sent unchanged in the response, and is not used by the POSLynx.
ClientMAC	Y - if requests are sent directly to POSLynx.	String	12	MAC Address of client. This id must be registered on POSLynx. This is added automatically by Precidia 'helper' applications (TNP-CG/TNP-NP), so is not always required. If requests are sent directly to POSLynx this is required.
Amount	Y	Float	1-7	Amount of transaction. Two decimal places, no dollar sign – (e.g. 123.45) Positive values only.
IPAddress	N	String	7-15	Override the configured POSLynx IP Address. Allows controlling which POSLynx processes each transaction individually. Only applicable to TNP-CG.
IPPort	N	Numeric	4-5	Override the configured POSLynx port. Used with *<IPAddress>* to control which POSLynx and port are used to process each transaction. Only applicable to TNP-CG.
TranDetail	N	TAG	-	See section 7 - TranDetail can be used to send detailed information about the particular transaction for detailed reporting.

Table 49

Common Elements for Responses

Tags common to all cash transaction responses, are outlined in the table below.

Element Tag	Req	Type	Size	Notes
Result	Y	Fixed String	-	One of APPROVED / ERROR
Command	Y	Fixed String	-	Command returned is same as sent on the request.
ResultText	Y	String	1-20	Text to describe result – e.g. explain error or result that could be placed on POS screen.
RecNum	Y - if approved	String	4	Reference number used in POSLynx to refer to this transaction. This is not as important for cash as it is for other transactions – since it is not possible to void the cash transaction.
ErrorCode	Y	String	4	Error code - 0000 if everything is successful. A list of error codes is yet to be defined.
Id	N	String	1-20	Optional Id used by POS to track the transaction e.g. invoice # or receipt #. Sent unchanged in response. If sent in request it will be returned in response.
ClientId	N	String	1-20	The ID of the user/cashier operating the PINpad. Sent unchanged in the response, and is not used by the POSLynx.
CardNumber	Y	String	4	For Cash '0000' is always returned. There is no card number.
CardType	Y	Fixed String	-	Type of card sent in request. Will display 'OtherCard' for cash. Based on card ranges defined in POSLynx configuration.

Table 50

Cash Sale

This transaction processes a cash sale. This transaction is handled completely by the POS, with the POSLynx recording the amount for statistics purposes. This transaction is sent when a customer makes a cash purchase.

Element Tag	Required	Type	Size	Notes
Command	Y	Fixed String	-	CASHSALE

Table 51

Example Request

A cash sale transaction:

```
<PLRequest>
    <Command>CASHSALE</Command>
    <Amount>5.00</Amount>
    <Id>9999</Id>
</PLRequest>
```

Example Response

A successful transaction response:

```
<PLResponse>
    <Result>APPROVED</Result>
    <CardNumber>0000</CardNumber>
    <CardType>OtherCard</CardType>
    <AuthAmt>5.00</AuthAmt>
    <RecNum>1111</RecNum>
    <Command>CASHSALE</Command>
    <ResultText>Transaction Approved</ResultText>
    <ErrorCode>0000</ErrorCode>
    <Id>9999</Id>
</PLResponse>
```

Cash Refund

This transaction processes a cash refund. This transaction is handled completely by the POS, with the POSLynx recording the amount for statistics purposes. This transaction is sent when a customer is given a cash refund.

Element Tag	Required	Type	Size	Notes
Command	Y	Fixed String	-	CASHREFUND

Table 52

Example Request

A cash refund transaction:

```
<PLRequest>
    <Command>CASHREFUND</Command>
    <Amount>2.00</Amount>
    <Id>9999</Id>
</PLRequest>
```

Example Response

A successful transaction response:

```
<PLResponse>
    <Result>APPROVED</Result>
    <CardNumber>0000</CardNumber>
    <CardType>OtherCard</CardType>
    <AuthAmt>2.00</AuthAmt>
    <RecNum>1111</RecNum>
    <Command>CASHREFUND</Command>
    <ResultText>Transaction Approved</ResultText>
    <ErrorCode>0000</ErrorCode>
    <Id>9999</Id>
</PLResponse>
```

Payroll Check Transactions

Payroll check processing allows a customer to enroll when cashing the first check, allowing easier subsequent cashing of checks (since the customer is already known).

Specific information on each transaction type is found in the following sections. Information on common element tags for requests and responses and their possible values, is contained in the two (2) tables that follow.

This information is not repeated for each transaction type. Only transaction-specific elements are detailed for each transaction type.

Common Elements for Requests

Tags common to all request transactions are outlined in the table below.

Element Tag	Req	Type	Size	Notes
Command	Y	Fixed String	-	Specifies transaction type to be executed. Must be one of: PAYENROLL / PAYREPEAT
Id	N	String	1-20	Optional Id used by the POS system to track the transaction e.g. invoice # or receipt #. Sent unchanged in response.
ClientId	N	String	1-20	The ID of the user/cashier operating the PINpad. Sent unchanged in the response, and is not used by the POSLynx.
ClientMAC	Y - if requests are sent directly to POSLynx.	String	12	MAC Address of client. This id must be registered on POSLynx. This is added automatically by Precidia 'helper' applications (TNP-CG/TNP-NP), so is not always required. If requests are sent directly to POSLynx this is required.
IPAddress	N	String	7-15	Override the configured POSLynx IP Address. Allows controlling which POSLynx processes each transaction individually. Only applicable to TNP-CG.
IPPort	N	Numeric	4-5	Override the configured POSLynx port. Used with *<IPAddress>* to control which POSLynx and port are used to process each transaction. Only applicable to TNP-CG.
TranDetail	N	TAG	-	See section 7 - TranDetail can be used to send detailed information about the particular transaction for detailed reporting.

Table 53

Common Elements for Responses

Tags common to all response transactions are outlined in the table below.

Element Tag	Required	Type	Size	Notes
Result	Y	Fixed String	-	One of APPROVED / DECLINED / ERROR
Command	Y	Fixed String	-	Command returned is same as sent on the request.
ResultText	Y	String	1-20	Text to describe result – e.g. explain error or result that could be placed on POS screen.
Authorization	Y - If approved	String	1-20	Authorization code from processor.
RecNum	Y - If approved	String	4	Reference number used to refer to this transaction in future transactions (e.g. to void this transaction, RecNum should be used).
RefData	N	Int	1-20	Additional reference data returned by host. Used only by certain hosts.
ErrorCode	Y	String	4	Error code - 0000 if everything is successful. 1 if Result is DECLINED. Numeric error code if the Result is ERROR.
Id	N	String	1-20	Optional Id used by the POS system to track the transaction (e.g. invoice # or receipt #). Sent unchanged in response. If sent in request it will be returned in response.
ClientId	N	String	1-20	The ID of the user/cashier operating the PINpad. Sent unchanged in the response, and is not used by the POSLynx.
AuthAmt	Y – if Approved	Float	3-7	Amount the transaction was approved for.
MerchantId	N	String	1-20	Merchant Id used to process transaction with host
TerminalId	N	String	1-20	Terminal Id used to process transaction with host
Receipt / Receipt#	N			If enabled in POSLynx, a printable host dependent Receipt will be part of the response. Not shown in examples, see section 6.

Table 54

Payroll Check Enrollment Transaction

This transaction enrolls a customer's paycheck with the host processor.

Driver's licenses can be swiped or manually entered. Checks should be scanned and MICR data sent. Separate fields are given for the check data (Sequence No, Account No, and Routing No), although this may permitted in all situations.

Element Tag	Required	Type	Size	Notes
Command	Y	Fixed String	-	PAYENROLL
Amount	Y	Float	1-7	Amount of transaction. Two decimal places, no dollar sign – (e.g. 123.45) Positive values only.
DLInput	Y	Fixed String	-	Specifies type of card data given. Must be one of : SWIPED/MANUAL/EXTERNAL
DLTrack	Y - If Swiped	String	1-20	Drivers License Swiped track 1 & track 2 data.Required if SWIPED
LicenseNo	Y – If Manual	String	1-20	Drivers License Account number. Required if MANUAL.
State	Y – If Manual	String	2	US State of person cashing Check. Required if MANUAL.
DOB	Y - If Manual	Int	8	Date of Birth of person cashing Check in format MMDDYYYY
SSN	Y	Int	9	Social Security Number of person cashing check - 9 digits.
LastName	Y	String	1-20	Last Name of person cashing check.
CheckDate	Y	Int	8	Issue Date on Check MMDDYYYY
CheckInput	Y	Fixed String	-	Specifies type of check data given. Must be one of : SWIPED/MANUAL/EXTERNAL
CheckMICR	Y – If Swiped	String	1-20	MICR line from the Check. RAW TOAD formatting is accepted. Typically data read from check reader can be passed in this field for processing.
CheckSeqNo	Y - If Manual	Int	12	Check Sequence Number – from MICR line.
CheckAcctNo	Y - If Manual	Int	26	Check Account Number – from MICR line.
CheckRouteNo	Y - If Manual	Int	9	Check Routing Number – from MICR line.

Table 55

Example Request

A check transaction with swiped Driver's License:

```
<PLRequest>
    <Command>PAYENROLL</Command>
    <Amount>123.45</Amount>
    <DLInput>SWIPED</DLInput>
    <DLTrack>%NTNEWYORK^FOX$SAM$J^12 ABC ST^?;63602009638527=0798=?</DLTrack>
    <SSN>111222333</SSN>
    <LastName>Smith</LastName>
    <CheckDate>12202009</CheckDate>
    <CheckInput>SWIPED</CheckInput>
    <CheckMICR>o1973ot123101092t23622o</CheckMICR>
    <Id>9999</Id>
</PLRequest>
```

A manually keyed drivers license and check:

```
<PLRequest>
    <Command>PAYENROLL</Command>
    <Amount>123.45</Amount>
    <DLInput>MANUAL</DLInput>
    <LicenseNo>869258149</LicenseNo>
    <DOB>10201965</DOB>
    <State>FL</State>
    <SSN>111222333</SSN>
    <LastName>Smith</LastName>
    <CheckDate>12202009</CheckDate>
    <CheckInput>MANUAL</CheckInput>
    <CheckSeqNo>1234</CheckSeqNo>
    <CheckAcctNo>123456789</CheckAcctNo>
    <CheckRouteNo>12345</CheckRouteNo>
    <Id>9999</Id>
</PLRequest>
```

Example Response

An approved transaction response:

```
<PLResponse>
```

```
    <Result>APPROVED</Result>
    <Command>PAYENROLL</Command>
    <ResultText>Transaction Approved</ResultText>
    <Authorization>123456</Authorization>
    <AuthAmt>123.45</AuthAmt>
    <RefData>0000</RefData>
    <RecNum>1015</RecNum>
    <ErrorCode>0000</ErrorCode>
    <CardType>Check</CardType>
    <MerchantId>1234567890</MerchantId>
    <TerminalId>012</TerminalId>
    <Id>9999</Id>
</PLResponse>
```

Payroll Check Repeat Transaction

This transaction performs a paycheck processing transaction for a previously enrolled customer.

Checks should be scanned and MICR data sent. Separate fields are given for check data (Sequence No, Account No, and Routing No), although this may permitted in all situations.

Element Tag	Required	Type	Size	Notes
Command	Y	Fixed String	-	PAYREPEAT
Amount	Y	Float	1-7	Amount of transaction. Two decimal places, no dollar sign – (e.g. 123.45) Positive values only.
SSN	Y	Int	9	Social Security Number of person cashing check 9 digits.
CheckDate	Y	Int	8	Issue Date on Check MMDDYYYY
CheckInput	Y	Fixed String	-	Specifies type of check data given. Must be one of : SWIPED/MANUAL/EXTERNAL
CheckMICR	Y – If Swiped	String	1-20	MICR line from the Check.
CheckSeqNo	Y - If Manual	Int	12	Check Sequence Number – from MICR line.
CheckAcctNo	Y - If Manual	Int	26	Check Account Number – from MICR line.
CheckRouteNo	Y - If Manual	Int	9	Check Routing Number – from MICR line.
LastName	N	String	1-20	Last Name of person cashing check.

Table 56

Example Request

A check transaction with a swiped Driver's License:

```
<PLRequest>
    <Command>PAYREPEAT</Command>
    <Amount>123.45</Amount>
    <SSN>111222333</SSN>
    <CheckDate>12202009</CheckDate>
    <CheckInput>SWIPED</CheckInput>
    <CheckMICR>o1973ot123101092t23622o</CheckMICR>
    <Id>9999</Id>
</PLRequest>
```

A manually keyed Driver's License and check:

```
<PLRequest>
    <Command>PAYREPEAT</Command>
    <Amount>123.45</Amount>
    <SSN>111222333</SSN>
    <CheckDate>12202009</CheckDate>
    <CheckInput>MANUAL</CheckInput>
    <CheckSeqNo>1234</CheckSeqNo>
    <CheckAcctNo>123456789</CheckAcctNo>
    <CheckRouteNo>12345</CheckRouteNo>
    <Id>9999</Id>
</PLRequest>
```

Example Response

An approved transaction response:

```
<PLResponse>
    <Result>APPROVED</Result>
    <Command>PAYENROLL</Command>
    <ResultText>Transaction Approved</ResultText>
    <Authorization>123456</Authorization>
    <AuthAmt>123.45</AuthAmt>
    <RefData>0000</RefData>
    <RecNum>1015</RecNum>
    <ErrorCode>0000</ErrorCode>
    <CardType>Check</CardType>
    <MerchantId>1234567890</MerchantId>
    <TerminalId>012</TerminalId>
```

```
    <Id>9999</Id>
</PLResponse>
```

Check Verification/Conversion Transactions

**The information in this section is in a DRAFT state and may be subject to change.

A merchant can accept checks which are verified in many ways. The POSLynx has various settings controlling how the host will process check transactions.

Specific information on each transaction type is found in the following sections. Information on common element tags for requests and responses and their possible values, is contained in the two (2) tables that follow.

This information is not repeated for each transaction type. Only transaction-specific elements are discussed for each transaction type.

Common Elements for Requests

Tags common to all request transactions are outlined in the table below.

Element Tag	Req	Type	Size	Notes
Command	Y	Fixed String	-	CHKVERIFY / CHKCONVERT
Id	N	String	1-20	Optional Id used by the POS system to track the transaction e.g. invoice # or receipt #. Sent unchanged in response.
ClientId	N	String	1-20	The ID of the user/cashier operating the PINpad. Sent unchanged in the response, and is not used by the POSLynx.
ClientMAC	Y - if requests are sent directly to POSLynx.	String	12	MAC Address of client. This id must be registered on POSLynx. This is added automatically by Precidia 'helper' applications (TNP-CG/TNP-NP), so is not always required. If requests are sent directly to POSLynx this is required.
IPAddress	N	String	7-15	Override the configured POSLynx IP Address. Allows controlling which POSLynx processes each transaction individually. Only applicable to TNP-CG.
IPPort	N	Numeric	4-5	Override the configured POSLynx port. Used with *<IPAddress>* to control which POSLynx and port are used to process each transaction. Only applicable to TNP-CG.
TranDetail	N	TAG	-	See section 7 - TranDetail can be used to send detailed information about the particular transaction for detailed reporting.

Table 57

Common Elements for Responses

Tags common to all response transactions are outlined in the table below.

Element Tag	Req	Type	Size	Notes
Result	Y	Fixed String	-	One of APPROVED / DECLINED / ERROR
Command	Y	Fixed String	-	Command returned is same as sent on the request.
ResultText	Y	String	1-20	Text to describe result – e.g. explain error or result that could be placed on POS screen.
Authorization	Y - If approved	String	1-20	Authorization code from processor.
RecNum	Y - If approved	String	4	Reference number used to refer to this transaction in future transactions (e.g. to void this transaction, RecNum should be used).
RefData	N	Int	1-20	Additional reference data returned by host – may not be used – or may only be used by certain hosts.
ErrorCode	Y	String	4	Error code: 0000 = Success. 1 = DECLINED. Numeric error code if the Result is ERROR.
Id	N	String	1-20	Optional Id used by the POS system to track the transaction (e.g. invoice # or receipt #).Sent unchanged in response. If sent in request it will be returned in response.
ClientId	N	String	1-20	The ID of the user/cashier operating the PINpad. Sent unchanged in the response, and is not used by the POSLynx.
AuthAmt	Y – if Approved	Float	3-7	Amount that transaction was approved for.
MerchantId	N	String	1-20	Merchant Id used to process transaction with host
TerminalId	N	String	1-20	Terminal Id used to process transaction with host
Receipt / Receipt#	N			If enabled in POSLynx, a printable host dependent Receipt will be part of the response. Not shown in examples, see section 6.

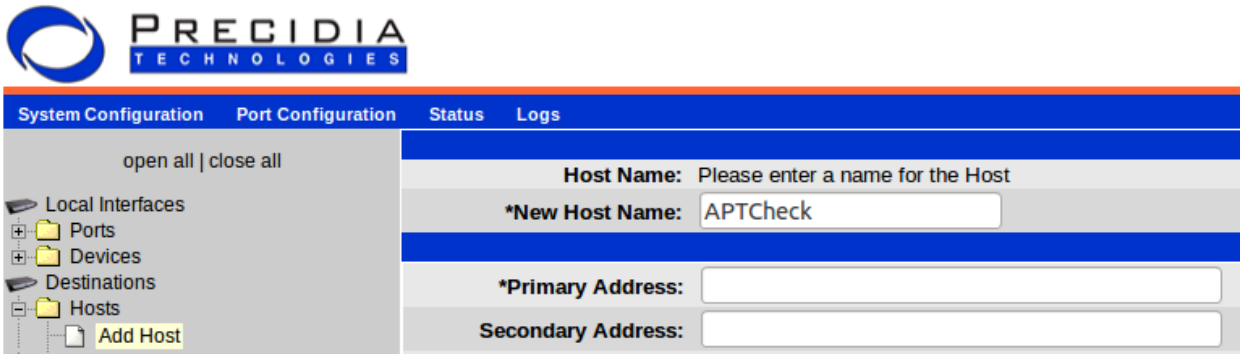
Table 58

Check Verify Transaction

Verification is run on a check transaction before it is submitted electronically for processing. Currently, this transaction is only supported for the APT XWeb host. The verification compares the checking account against a list of 'bad' checking account numbers. If there is no match, the transaction is 'approved'.

If there is a match, the transaction is 'declined'. Verification reduces but does not eliminate risk of returned checks for merchants.

NOTE: In the POSLynx GUI, the value in the 'New Host Name' field must be provisioned as "APTCheck". This value is case sensitive. See screenshot below.



Request Elements:

The following fields must be submitted along with the Header Fields for this type of transaction:

Element Tag	Req	Type	Size	Notes
Command	Y	Fixed String	-	CHKVERIFY
Amount	Y	Float	1-7	0.01 - 99999.99 (decimal point not required for even dollar amounts).
RoutingNumber	Conditional	Numeric	9	When input is MANUAL. If RoutingNumber is used, AccountNumber must be used as well. RoutingNumber cannot be used with MICR.
CheckNumber	N	Numeric	1-15	CheckNumber cannot be used with SWIPED. CheckNumber will be system generated if the check number is not available.
DLNumber	N	Alphanumeric	Variable	If provided, it must contain the 2 character state abbreviation followed by a hyphen, then the actual license number. For example: AZ-D123456 means Arizona driver's license number D123456.

AccountNumber	Conditional	Numeric	1-15	When input is MANUAL, and AccountNumber is used, RoutingNumber must be used as well. AccountNumber cannot be used with MICR.
MICR	Conditional	Alphanumeric	Variable	When Input is SWIPED, enter the Code read from MICR reader. See conversions for special conversion codes from various reader types.
Input	Y	String		MANUAL if CheckNumber, Routing Number and AccountNumber is inputted, or SWIPED if MICR presented instead.

Table 59

Example Requests

Scanned MICR:

```
<PLRequest>
    <Command>CHKVERIFY</Command>
    <Amount>1.00</Amount>
    <DLNumber>CA-D1234567</DLNumber>
    <MICR>c002096c d081015218d 5030337501c</MICR>
    <Input>SWIPED</Input>
    <Id>9999</Id>
</PLRequest>
```

Manually entered check:

```
<PLRequest>
      <Command>CHKVERIFY</Command>
      <Amount>1.00</Amount>
      <RoutingNumber>112000066</RoutingNumber>
      <CheckNumber>9999</CheckNumber>
      <DLNumber>CA-D1234567</DLNumber>
      <AccountNumber>12121</AccountNumber>
      <Input>MANUAL</Input>
      <Id>9999</Id>

</PLRequest>
```

Response Elements:

Element Tag	Req	Type	Size	Notes
Error Code	Y	Numeric	3	0000=success. See below for full list of Error Codes.
ResultText	Y	Alphanumeric	Variable	Gives a more detailed description on the ResponseCode.
Authorization	Conditional	Alphanumeric	6	Provides the verification code for successful check verifications.

RefData	Y	Numeric	12	Host processor reference number.
RecNum	Y	Numeric	12	Used for managing the record on MerchantVu.
InputMethod	N	String	1-20	How the card was entered. One of SWIPED/MANUAL/TOKEN/CHIP/TAP/RFID. See 'Card Entry Options' in Chapter 5 for more detail.

Table 57

Example Response

```
<PLResponse>
      <Result>APPROVED</Result>
      <Authorization>790-105</Authorization>
      <RecNum />
      <RefData>000000000047</RefData>
      <ErrorCode>0000</ErrorCode>
      <AuthAmt>1.00</AuthAmt>
      <TransactionDate>130919</TransactionDate>
      <TransactionTime>160021</TransactionTime>
      <Command>CHKVERIFY</Command>
      <ResultText>Check Approval</ResultText>
      <Id>9999</Id>
</PLResponse>
```

Error Codes

Error Code	Description
011	CheckSubmitted – Returned for successful check sale or check credit transaction.
012	CheckApproval
013	CheckDecline
014	CheckWarning
015	CheckError
016	Check Transaction Amount Limit Exceeded
017	Check Daily Amount Limit Exceeded
018	Check Monthly Amount Limit Exceeded
019	RDFI Not Qualified to Participate
020	Corporate Customer Advises Not Authorized
021	Check Not Previously Authorized
022	Ineligible Transaction For ACH Network
207	Invalid User Credentials
208	User Locked Out
209	Security Question/Answer/Password Not Set
210	Password Expired
212	Temporary Password From Reset Needs To Be Updated
800-899	Gateway Errors

900-999	Processor Errors

Table 58

NOTES:

Routing #: start and end with "d"

Account #: end with a "c"

Check #: no characters added

Acceptable Formats:

ROUTING # - ACCOUNT # - CHECK #

d081015218d 5030337501c 002096

ROUTING # - ACCOUNT #

d081015218d 5030337501c

We've also found that it works if you encapsulate the check number with a "c" at the start and end of the check number and have it at the start of the MICR string.

Acceptable Format(s):

CHECK # ROUTING # ACCOUNT #

c002096c d081015218d 5030337501c

The account number has to come before the routing number, the check number can be at the beginning or end.

Check Conversion Transaction

This command performs a conversion to an ACH/EFT transaction.

The POSLynx can be configured for 3 different modes. Depending on the mode used, different data may be required to be sent.

Full MICR 1 (FM1)

The MICR information can be manually entered data (Check Serial No, Bank Routing No, and Account No), but it must be swiped for conversion to occur. The check amount is required. The customer's 10-digit phone number may be required.

Full MICR 2 (FM2)

The MICR information can be manually entered data (Check Serial No, Bank Routing No, and Account No), but it must be swiped for conversion to occur. The customer's Driver's License information is required, either swiped or manually entered. The State/Province and Date of Birth in addition to the check amount is required. The customer's 10 digit phone number may be required as well.

ID Free (NID)

The MICR information can be manually entered data (Check Serial No, Bank Routing No, and Account No), but it must be swiped for conversion to occur. The check amount is required. The customer's 10-digit phone number may be required as well. If the host does not find a match in the cross reference file, it will request Driver's License information, either swiped or manually. Finally, State/Province and Date of Birth are required.

Conversion Only

There is a POSLynx check setting for ECC Conversion only. This setting is available for any of the above settings (FM1, FM2, NID). This setting requires that the MICR is swiped. Other information can be provided, but is not verified. The host does not warranty the transaction, it is a settlement-only service.

Element Tag	Required	Type	Size	Notes
Command	Y	Fixed String	-	CHKCONVERT
Amount	Y	Float	1-7	Amount of transaction. Two decimal places, no dollar sign – (e.g. 123.45) Positive values only.
DLInput	Y	Fixed String	-	Specifies type of card data given. Must be one of : SWIPED/MANUAL/EXTERNAL
DLTrack	Y - If Swiped	String	1-20	Drivers License Swiped track 1 & track 2 data.Required if SWIPED
LicenseNo	Y – If Manual	String	1-20	Drivers License Account number.Required if MANUAL.
State	Y – If Manual	String	2	US State of person cashing Check.Required if MANUAL.
DOB	Y - If Manual	Int	8	Date of Birth of person cashing Check in format MMDDYYYY

CheckDate	Y	Int	8	Issue Date on Check MMDDYYYY
CheckInput	Y	Fixed String	-	Specifies type of check data given.Must be one of : SWIPED/MANUAL/EXTERNAL
CheckMICR	Y – If Swiped	String	1-20	MICR line from the Check. RAW TOAD formatting is accepted. Typically data read from check reader can be passed in this field for processing.
CheckSeqNo	Y - If Manual	Int	12	Check Sequence Number – from MICR line.
CheckAcctNo	Y - If Manual	Int	26	Check Account Number – from MICR line.
CheckRouteNo	Y - If Manual	Int	9	Check Routing Number – from MICR line.
PhoneNumber	N	Int	10	Phone number of the consumer.

Table 59

Example Request

A basic check conversion transaction:

```
<PLRequest>
    <Command>CHKCONVERT</Command>
    <Amount>123.45</Amount>
    <CheckDate>12202009</CheckDate>
    <CheckInput>SWIPED</CheckInput>
    <CheckMICR>T123458210t1111392100o 12329</CheckMICR>
    <PhoneNumber>9998881234</PhoneNumber>
    <Id>9999</Id>
</PLRequest>
```

A check conversion transaction with a swiped Driver's License:

```
<PLRequest>
    <Command>CHKCONVERT</Command>
    <Amount>123.45</Amount>
    <CheckDate>12202009</CheckDate>
    <DLInput>SWIPED</DLInput>
    <DLTrack>%NTNEWYORK^FOX$SAM$J^12 ABC ST^?;6360200096527=07198=?</DLTrack>
    <CheckInput>SWIPED</CheckInput>
    <CheckMICR>T123458210t1111392100o 12329</CheckMICR>
    <PhoneNumber>9998881234</PhoneNumber>
    <Id>9999</Id>
</PLRequest>
```

A manually keyed Driver's License and check:

```
<PLRequest>
    <Command>CHKCONVERT</Command>
    <Amount>123.45</Amount>
    <CheckDate>12202009</CheckDate>
    <DLInput>MANUAL</DLInput>
    <LicenseNo>869258149</LicenseNo>
    <DOB>10201965</DOB>
    <State>FL</State>
    <CheckInput>MANUAL</CheckInput>
    <CheckSeqNo>1234</CheckSeqNo>
    <CheckAcctNo>123456789</CheckAcctNo>
    <CheckRouteNo>12345</CheckRouteNo>
    <PhoneNumber>9998881234</PhoneNumber>
    <Id>9999</Id>
</PLRequest>
```

Example Response

An approved transaction response:

```
<PLResponse>
    <Result>APPROVED</Result>
    <Command>CHKCONVERT</Command>
    <ResultText>Transaction Approved</ResultText>
    <Authorization>123456</Authorization>
    <AuthAmt>123.45</AuthAmt>
    <RefData>0000</RefData>
    <RecNum>1015</RecNum>
    <ErrorCode>0000</ErrorCode>
    <CardType>Check</CardType>
    <MerchantId>1234567890</MerchantId>
    <TerminalId>012</TerminalId>
    <Id>9999</Id>
</PLResponse>
```

Converting Check MICR from Readers

This section describes the way the check MICR is converted based on different readers.

Reader Type	From	To	From	To	From	To	From	To	From	To
Digital Check	":"	"d”	"<"	"c"	";"	"b"	"="	"-"	"?"	"!"
RDM	"T"	"d"	"O"	"c"	"A"	"b"	"D"	"-"		
MagTek	"T"	"d"	"U"	"c"	"$"	"b"				

Table 60

Example of good MICR from an RDM reader:

O0001053O T123456780T 123456789O

Steps to follow for conversion:
1. Convert MICR line using conversion chart above.
Example: O0001053O T123456780T 123456789O

2. Using the spaces, parse the line into three fields.
For example:

   a. O0001053O
   b. T123456780T
   c. 123456789O

3. For the first field, make the first and last characters a lower case "c".
   a. Example: c001053c

4. For the second field, make the first and last characters a lower case "d".
   a. Example: d123456780d

5. For the third field, make the last character a lower case "c".
   a. Example: 123456789c

6. Put the MICR line back together using these three fields.
   a. Example: <MICR>c001053c d123456780d 123456789c</MICR>

Electronic Benefits (EBT) Transactions

Electronic Benefits Transactions (EBT) are supported on the POSLynx, although not all hosts supporting US Debit support EBT. The Verifone 1000SE and SC5000 PIN Pads are currently supported. The PIN Pad can be connected to the either a POSLynx serial port or to the serial port of the POS system. Multiple PIN Pad devices can be controlled by connecting the PIN Pads to the network through a Precidia *iPocket* device.

When an XML request is made, the POSLynx will control the PIN Pad and prompt the user for a PIN, which is used to confirm the transaction with the host processor. If the PIN Pad is connected to the POS system, then one of the Precidia helper applications must be used to control the PIN Pad.

Specific information on each transaction type is found in the following sections. Information on common element tags for requests and responses and their possible values, is contained in the two (2) tables that follow.

This information is not repeated for each transaction type. Only transaction-specific elements are discussed for each transaction type.

Common Elements for Requests

Tags common to all request transactions are outlined in the table below.

Element Tag	Required	Type	Size	Notes
Command	Y	Fixed String	-	Specifies transaction type to be executed. Must be one of: EBTFOODSALE / EBTFOODREFUND / EBTCASHSALE / EBTVOID / EBTWITHDRAWAL EBTFOODVOUCHERSALE / EBTFOODVOUCHERREFUND / EBTCASHVOUCHERSALE / EBTFOODBALANCE / EBTCASHBALANCE
Id	N	String	1-20	Optional Id used by POS to track the transaction e.g. invoice # or receipt #. Sent unchanged in response, it is not used by POSLynx.
ClientId	N	String	1-20	The ID of the user/cashier operating the PINpad. Sent unchanged in the response, and is not used by the POSLynx.
ClientMAC	Y - if requests are sent directly to POSLynx.	String	12	MAC Address of client. This id must be registered on POSLynx. This is added automatically by Precidia 'helper' applications (TNP-CG/TNP-NP), so is not always required. If requests are sent directly to POSLynx this is required.
Amount	Y	Float	1-7	Amount of transaction. Two decimal places, no dollar sign – (e.g. 123.45) Positive values only.
Input	Y	Fixed String	-	Specifies type of card data given. Must be one of SWIPED/MANUAL/EXTERNAL.
Track2	Y - If	String	1-40	Swiped track2 data - without sentinel

	Swiped			characters. Required if SWIPED
Track1	N	String	1-80	Swiped track1 data - without start/end sentinel characters.
IPAddress	N	String	7-15	Override the configured POSLynx IP Address. Allows controlling which POSLynx processes each transaction individually. Only applicable to TNP-CG.
IPPort	N	Int	4-5	Override the configured POSLynx port. Used with *<IPAddress>* to control which POSLynx and port are used to process each transaction. Only applicable to TNP-CG.
TranDetail	N	TAG	-	See section 7 - TranDetail can be used to send detailed information about the particular transaction for detailed reporting.

Table 61

Common Elements for Responses

Tags common to all response transactions are outlined in the table below.

Element Tag	Required	Type	Size	Notes
Result	Y	Fixed String	-	One of APPROVED / DECLINED / ERROR
Command	Y	Fixed String	-	Command returned is same as sent on the request.
ResultText	Y	String	1-20	Text to describe result – e.g. explain error or result that could be placed on POS screen.
Authorization	Y - if approved	String	1-20	Authorization code from processor.
RecNum	Y - if approved	String	4	Reference number used to refer to this transaction in future transactions. e.g. to void this transaction, RecNum should be used.
RefData	N	Int	1-20	Additional reference data returned by host – may not be used – or may only be used by certain hosts.
ErrorCode	Y	String	1-20	Error code - 0000 if everything is successful. A list of error codes is yet to be defined.
Id	N	String	1-20	Optional Id used by POS to track the transaction e.g. invoice # or receipt #. Sent unchanged in response. If sent in request it will be returned in response.
ClientId	N	String	1-20	The ID of the user/cashier operating the PINpad. Sent unchanged in the response, and is not used by the POSLynx.
CardNumber	Y	String	4	Returns last four digits of card number – for display if needed.
CardType	Y	Fixed String	-	Type of card sent in request. Will display 'OtherCard' for debit. Based on card ranges defined in POSLynx configuration.
MerchantId	N	String	1-20	Merchant Id used to process transaction with

				host
TerminalId	N	String	1-20	Terminal Id used to process transaction with host
Receipt / Receipt#	N			If enabled in POSLynx, a printable host dependant Receipt will be part of the response. Not shown in examples, see section 6.
InputMethod	N	String	1-20	How the card was entered. One of SWIPED/MANUAL/TOKEN/CHIP/TAP/RFID. See 'Card Entry Options' in Chapter 5 for more detail.

Table 62

EBT Food Stamp Sale

This transaction processes an EBT Food Stamp sale. This command removes the transaction amount from the customer account, and will increase the batch amount for the merchant. The card holder will be prompted for a PIN number on the PIN Pad.

Request Elements

Element Tag	Required	Type	Size	Notes
Command	Y	Fixed String	-	EBTFOODSALE
Input	Y	Fixed String	-	Specifies type of card data given. Must be one of : SWIPED/EXTERNAL/TOKEN

Table 63

NOTE: Manual card entry is not allowed.

Example Request

A transaction with Track2 data specified. The PIN Pad will prompt for PIN entry only:

```
<PLRequest>
    <Command>EBTFOODSALE</Command>
    <Amount>2.00</Amount>
    <Input>SWIPED</Input>
    <Track2>9988999800002773=1012101543211234</Track2>
    <Track1>9988999800002773^LAST/FIRST^10121015432112345678</Track1>
    <Id>9999</Id>
</PLRequest>
```

A transaction specifying an EXTERNAL swipe. The PIN Pad will prompt for both the swipe and PIN entry:

```
<PLRequest>
    <Command>EBTFOODSALE</Command>
    <Amount>2.00</Amount>
    <Input>EXTERNAL</Input>
    <Id>9999</Id>
</PLRequest>
```

Response Elements

Element Tag	Required	Type	Size	Notes
AuthAmt	Y	Float	3-7	Amount processed and removed from balance
FoodBalance	Y	Float	3-7	Remaining Food Stamp balance
CashBalance	N	Float	3-7	Remaining Cash Balance
InputMethod	N	String	1-20	How the card was entered. One of SWIPED/TOKEN/CHIP/TAP/RFID. See 'Card Entry Options' in Chapter 5 for more detail.

Table 64

Example Response

An approved transaction response:

```
<PLResponse>
    <Command>EBTFOODSALE</Command>
    <Result>APPROVED</Result>
    <CardNumber>2773</CardNumber>
    <CardType>EBT</CardType>
    <Authorization>091236</Authorization>
    <AuthAmt>2.00</AuthAmt>
    <RecNum>1001</RecNum>
    <RefData>00000001</RefData>
    <FoodBalance>76.00</FoodBalance>
    <CashBalance>78.00</CashBalance>
    <ResultText>Transaction Approved</ResultText>
    <MerchantId>1234567890</MerchantId>
    <TerminalId>012</TerminalId>
    <ErrorCode>0000</ErrorCode>
    <Id>9999</Id>
    <InputMethod>SWIPED</InputMethod>
</PLResponse>
```

EBT Food Stamp Refund

This transaction processes an EBT Food Stamp refund. This command adds the transaction amount to the customer account, and will decrease the batch amount for the merchant. The card holder will be prompted for a PIN number on the PIN Pad.

Request Elements

Element Tag	Required	Type	Size	Notes
Command	Y	Fixed String	-	EBTFOODREFUND
Input	Y	Fixed String	-	Specifies type of card data given. Must be one of : SWIPED/EXTERNAL/TOKEN

Table 65

Example Request

A transaction with Track2 data specified. The PIN Pad will prompt for PIN entry only:

```
<PLRequest>
    <Command>EBTFOODREFUND</Command>
    <Amount>2.00</Amount>
    <Input>SWIPED</Input>
    <Track2>9999999808802773=10121015432115678</Track2>
    <Id>9999</Id>
</PLRequest>
```

A transaction specifying an EXTERNAL swipe. The PIN Pad will prompt for both the swipe and PIN entry:

```
<PLRequest>
    <Command>EBTFOODREFUND</Command>
    <Amount>2.00</Amount>
    <Input>EXTERNAL</Input>
    <Id>9999</Id>
</PLRequest>
```

Response Elements

Element Tag	Required	Type	Size	Notes
AuthAmt	Y	Float	3-7	Amount processed and removed from balance
FoodBalance	Y	Float	3-7	Remaining Food Stamp balance
CashBalance	N	Float	3-7	Remaining Cash Balance
InputMethod	N	String	1-20	How the card was entered. One of SWIPED/TOKEN/CHIP/TAP/RFID. See 'Card Entry

				Options' in Chapter 5 for more detail.

Table 66

Example Response

An approved transaction response:

```
<PLResponse>
  <Result>APPROVED</Result>
  <CardNumber>2773</CardNumber>
  <CardType>EBT</CardType>
  <Authorization>093246</Authorization>
  <AuthAmt>2.00</AuthAmt>
  <RecNum>1003</RecNum>
  <RefData>00000002</RefData>
  <FoodBalance>76.00</FoodBalance>
  <CashBalance>78.00</CashBalance>
  <Command>EBTFOODREFUND</Command>
  <ResultText>Transaction Approved</ResultText>
  <MerchantId>1234567890</MerchantId>
  <TerminalId>001</TerminalId>
  <ErrorCode>0000</ErrorCode>
  <Id>9999</Id>
  <InputMethod>SWIPED</InputMethod>
</PLResponse>
```

EBT Cash Sale

This transaction processes an EBT Cash Sale. This command removes the transaction amount from the customer account, and will increase the batch amount for the merchant. The card holder will be prompted for a PIN number on the PIN Pad.

Request Elements

Element Tag	Required	Type	Size	Notes
Command	Y	Fixed String	-	EBTCASHSALE
Cashback	N	Float	3-7	Amount of extra cash to be added to total amount of transaction.
Input	Y	Fixed String	-	Specifies type of card data given. Must be one of : SWIPED/MANUAL/EXTERNAL/TOKEN

Table 67

Example Request

A transaction with Track2 data specified. The PIN pad will prompt for PIN entry only:

```
<PLRequest>
    <Command>EBTCASHSALE</Command>
    <Amount>15.00</Amount>
    <Cashback>5.00</Cashback>
    <Input>SWIPED</Input>
    <Track2>9998899800002773=10121015432345678</Track2>
    <Id>9999</Id>
</PLRequest>
```

A transaction specifying an EXTERNAL swipe. The PIN Pad will prompt for both the swipe and PIN entry:

```
<PLRequest>
    <Command>EBTCASHSALE</Command>
    <Amount>15.00</Amount>
    <Cashback>5.00</Cashback>
    <Input>EXTERNAL</Input>
    <Id>9999</Id>
</PLRequest>
```

Response Elements

Element Tag	Required	Type	Size	Notes
AuthAmt	Y	Float	3-7	Amount processed and removed from balance (equal to Amount + CashBack)
Amount	Y	Float	1-7	Amount of actual Sale
Cashback	N	Float	3-7	Amount of Cash given back to customer.
FoodBalance	Y	Float	3-7	Remaining Food Stamp balance
CashBalance	N	Float	3-7	Remaining Cash Balance
InputMethod	N	String	1-20	How the card was entered. One of SWIPED/MANUAL/TOKEN/CHIP/TAP/RFID. See 'Card Entry Options' in Chapter 5 for more detail.

Table 68

Example Response

An approved transaction response:

```
<PLResponse>
   <Result>APPROVED</Result>
   <CardNumber>2773</CardNumber>
   <CardType>EBT</CardType>
```

```
   <Authorization>293477</Authorization>
   <Amount>15.00</Amount>
   <Cashback>5.00</Cashback>
   <AuthAmt>20.00</AuthAmt>
   <RecNum>1007</RecNum>
   <RefData>00000019</RefData>
   <Command>EBTCASHSALE</Command>
   <ResultText>Transaction Approved</ResultText>
   <MerchantId>1234567890</MerchantId>
   <TerminalId>001</TerminalId>
   <ErrorCode>0000</ErrorCode>
   <Id>9999</Id>
   <InputMethod>SWIPED</InputMethod>
</PLResponse>
```

EBT Food Stamp Voucher Sale

This transaction processes an EBT Food Stamp Voucher. The Voucher Number is manually entered. No PIN is required.

Request Elements

Element Tag	Required	Type	Size	Notes
Command	Y	Fixed String	-	EBTFOODVOUCHERSALE
Voucher	Y	Int	1-10	Number from the voucher being processed.
VoiceAuth	Y	Int	1-10	Authorization number obtained by phone.
Input	Y	Fixed String	-	Specifies type of card data given. Must be MANUAL.

Table 69

Example Request

```
<PLRequest>
    <Command>EBTFOODVOUCHERSALE</Command>
    <Amount>2.00</Amount>
    <VoiceAuth>123456</VoiceAuth>
    <Voucher>12345678</Voucher>
    <CardNumber>9999999800002773</CardNumber>
    <ExpiryDate>1014</ExpiryDate>
    <Input>MANUAL</Input>
    <Id>9999</Id>
</PLRequest>
```

Response Elements

Element Tag	Required	Type	Size	Notes
AuthAmt	Y	Float	3-7	Amount processed and removed from balance
FoodBalance	Y	Float	3-7	Remaining Food Stamp balance
CashBalance	N	Float	3-7	Remaining Cash Balance
InputMethod	N	String	1-20	How the card was entered. Must be MANUAL.

Table 70

Example Response

An approved transaction response:

```
<PLResponse>
```

```
  <Result>APPROVED</Result>
  <CardNumber>2773</CardNumber>
  <CardType>EBT</CardType>
  <Authorization>123456</Authorization>
  <AuthAmt>2.00</AuthAmt>
  <RecNum>1008</RecNum>
  <RefData>00000017</RefData>
  <FoodBalance>76.00</FoodBalance>
  <CashBalance>78.00</CashBalance>
  <Command>EBTFOODVOUCHERSALE</Command>
  <ResultText>Transaction Approved</ResultText>
  <MerchantId>1234567890</MerchantId>
  <TerminalId>001</TerminalId>
  <ErrorCode>0000</ErrorCode>
  <Id>9999</Id>
  <InputMethod>MANUAL</InputMethod>
</PLResponse>
```

EBT Food Stamp Voucher Refund

This transaction processes an EBT Food Stamp Voucher Refund. The command reverses a Voucher Sale.

The Voucher Number is manually entered. No PIN is required.

Request Elements

Element Tag	Required	Type	Size	Notes
Command	Y	Fixed String	-	EBTFOODVOUCHERREFUND
Voucher	Y	Int	1-10	Number from the voucher being processed.
VoiceAuth	Y	Int	1-10	Authorization number obtained by phone.
Input	Y	Fixed String	-	Specifies type of card data given. Must be MANUAL.

Table 71

Example Request

A swiped transaction:

```
<PLRequest>
    <Command>EBTFOODVOUCHERREFUND</Command>
    <Amount>2.00</Amount>
    <VoiceAuth>123456</VoiceAuth>
    <Voucher>12345678</Voucher>
```

```
    <CardNumber>9999999800002773</CardNumber>
    <ExpiryDate>0114</ExpiryDate>
    <Input>MANUAL</Input>
    <Id>9999</Id>
</PLRequest>
```

Response Elements

Element Tag	Required	Type	Size	Notes
AuthAmt	Y	Float	3-7	Amount processed and removed from balance
FoodBalance	Y	Float	3-7	Remaining Food Stamp balance
CashBalance	N	Float	3-7	Remaining Cash Balance
InputMethod	N	String	1-20	How the card was entered. Must be MANUAL.

Table 72

Example Response

An approved transaction response:

```
<PLResponse>
   <Result>APPROVED</Result>
   <CardNumber>2773</CardNumber>
   <CardType>EBT</CardType>
   <Authorization>123456</Authorization>
   <AuthAmt>2.00</AuthAmt>
   <RecNum>1008</RecNum>
   <RefData>00000017</RefData>
   <FoodBalance>76.00</FoodBalance>
   <CashBalance>78.00</CashBalance>
   <Command>EBTFOODVOUCHERREFUND</Command>
   <ResultText>Transaction Approved</ResultText>
   <MerchantId>1234567890</MerchantId>
   <TerminalId>001</TerminalId>
   <ErrorCode>0000</ErrorCode>
   <Id>9999</Id>
   <InputMethod>MANUAL</InputMethod>
</PLResponse>
```

EBT Cash Voucher Sale

This transaction processes an EBT Cash Voucher Sale. The Voucher Number is manually entered. No PIN is required.

Request Elements

Element Tag	Required	Type	Size	Notes
Command	Y	Fixed String	-	EBTCASHVOUCHERSALE
Voucher	Y	Int	1-10	Number from the voucher being processed.
VoiceAuth	Y	Int	1-10	Authorization number obtained by phone.
Input	Y	Fixed String	-	Specifies type of card data given. Must be MANUAL.

Table 73

Example Request

A swiped transaction:

```
<PLRequest>
    <Command>EBTCASHVOUCHERSALE</Command>
    <Amount>2.00</Amount>
    <VoiceAuth>123456</VoiceAuth>
    <Voucher>12345678</Voucher>
    <CardNumber>9999999800002773</CardNumber>
    <ExpiryDate>0114</ExpiryDate>
    <Input>MANUAL</Input>
    <Id>9999</Id>
</PLRequest>
```

Response Elements

Element Tag	Required	Type	Size	Notes
AuthAmt	Y	Float	3-7	Amount processed and removed from balance
FoodBalance	N	Float	3-7	Remaining Food Stamp balance
CashBalance	Y	Float	3-7	Remaining Cash Balance
InputMethod	N	String	1-20	How the card was entered. Must be MANUAL.

Table 74

Example Response

An approved transaction response:

```
<PLResponse>
  <Result>APPROVED</Result>
  <CardNumber>2773</CardNumber>
  <CardType>EBT</CardType>
```

```
  <Authorization>123456</Authorization>
  <AuthAmt>2.00</AuthAmt>
  <RecNum>1009</RecNum>
  <RefData>00000001</RefData>
  <FoodBalance>76.00</FoodBalance>
  <CashBalance>78.00</CashBalance>
  <Command>EBTCASHVOUCHERSALE</Command>
  <ResultText>Transaction Approved</ResultText>
  <MerchantId>1234567890</MerchantId>
  <TerminalId>001</TerminalId>
  <ErrorCode>0000</ErrorCode>
  <Id>9999</Id>
  <InputMethod>MANUAL</InputMethod>
</PLResponse>
```

EBT Food Stamp Account Balance

This transaction obtains the balance of the EBT Food Stamp account. The card holder will be prompted for a PIN on the PIN Pad.

Both Food Stamp and cash balances may be returned for some hosts.

Request Elements

Element Tag	Required	Type	Size	Notes
Command	Y	Fixed String	-	EBTFOODBALANCE
Input	Y	Fixed String	-	Specifies type of card data given. Must be one of : SWIPED/MANUAL/EXTERNAL/TOKEN
Id	N	String	1-20	Used by the POS system to track the transaction e.g. invoice # or receipt #. Sent unchanged in response.

Table 75

Example Request

```
<PLRequest>
    <Command>EBTFOODBALANCE</Command>
    <Input>EXTERNAL</Input>
    <Id>9999</Id>
</PLRequest>
```

Response Elements

Element Tag	Required	Type	Size	Notes
AuthAmt	Y	Float	3-7	Amount processed and removed from balance
FoodBalance	Y	Float	3-7	Remaining Food Stamp balance

CashBalance	N	Float	3-7	Remaining Cash Balance
InputMethod	N	String	1-20	How the card was entered. One of SWIPED/MANUAL/TOKEN/CHIP/TAP/RFID. See 'Card Entry Options' in Chapter 5 for more detail.

Table 76

Example Response

An approved transaction response:

```
<PLResponse>
  <Result>APPROVED</Result>
  <CardNumber>2773</CardNumber>
  <CardType>EBT</CardType>
  <Authorization>095858</Authorization>
  <AuthAmt>0.00</AuthAmt>
  <RefData>00000002</RefData>
  <FoodBalance>76.00</FoodBalance>
  <CashBalance>78.00</CashBalance>
  <Command>EBTFOODBALANCE</Command>
  <ResultText>Transaction Approved</ResultText>
  <MerchantId>1234567890</MerchantId>
  <TerminalId>001</TerminalId>
  <ErrorCode>0000</ErrorCode>
  <Id>9999</Id>
  <InputMethod>EXTERNAL</InputMethod>
</PLResponse>
```

EBT Cash Account Balance

This transaction obtains a balance on the EBT Cash account. The card holder will be prompted for a PIN number on the PIN Pad.

Both Food Stamp and Cash Balances may be returned for some hosts.

Request Elements

Element Tag	Required	Type	Size	Notes
Command	Y	Fixed String	-	EBTCASHBALANCE
Input	Y	Fixed String	-	Specifies type of card data given. Must be one of : SWIPED/MANUAL/EXTERNAL/TOKEN

Table 77

Example Request

A swiped transaction:

```
<PLRequest>
    <Command>EBTCASHBALANCE</Command>
    <Input>EXTERNAL</Input>
    <Id>9999</Id>
</PLRequest>
```

Response Elements

Element Tag	Required	Type	Size	Notes
AuthAmt	Y	Float	3-7	Amount processed and removed from balance
FoodBalance	N	Float	3-7	Remaining Food Stamp balance
CashBalance	Y	Float	3-7	Remaining Cash Balance
InputMethod	N	String	1-20	How the card was entered. One of SWIPED/MANUAL/TOKEN/CHIP/TAP/RFID. See 'Card Entry Options' in Chapter 5 for more detail.

Table 78

Example Response

An approved transaction response:

```
<PLResponse>
  <Result>APPROVED</Result>
  <CardNumber>2773</CardNumber>
  <CardType>EBT</CardType>
  <Authorization>095957</Authorization>
  <AuthAmt>0.00</AuthAmt>
  <RefData>00000002</RefData>
  <FoodBalance>76.00</FoodBalance>
  <CashBalance>78.00</CashBalance>
  <Command>EBTCASHBALANCE</Command>
  <ResultText>Transaction Approved</ResultText>
  <MerchantId>1234567890</MerchantId>
  <TerminalId>001</TerminalId>
  <ErrorCode>0000</ErrorCode>
  <Id>9999</Id>
  <InputMethod>EXTERNAL</InputMethod>
</PLResponse>
```

EBT Void

This transaction removes a previous transaction from the merchant's batch, that is, it cancels the transaction. Typically, it is used to remove a charge from a customer when a problem is encountered with an EBT transaction.

Should the batch have been closed, an EBT Void transaction cannot be completed. In this case, an EBT Refund transaction must be performed.

This transaction is tied to the previous transaction via the *<RecNum>* element.

Request Elements

Element Tag	Required	Type	Size	Notes
Command	Y	Fixed String	-	EBTVOID
RecNum	Y	String	4	Reference to the original transaction to be found.
Amount	Y	Float	1-7	The original transaction amount
Input	Y	Fixed String	-	Specifies type of card data given. Must be one of : SWIPED/MANUAL/EXTERNAL/TOKEN
Id	N	String	1-20	Used by the POS system to track the transaction e.g. invoice # or receipt #. Sent unchanged in response.

Table 79

Example Request

A manual transaction:

```
<Input>MANUAL</Input>
    <Command>EBTVOID</Command>
    <RecNum>1234</RecNum>
    <CardNumber>9999999812000005</CardNumber>
    <Amount>5.00</Amount>
    <Id>9999</Id>
</PLRequest>
```

Response Elements

Element Tag	Required	Type	Size	Notes
Command	Y	Fixed String	-	EBTVOID
Result	Y	Fixed String	-	One of APPROVED / DECLINED / ERROR
RecNum	Y	String	4	Reference to the original transaction to be found.
Amount	Y	Float	1-7	The original transaction amount
InputMethod	N	String	1-20	How the card was entered. One of SWIPED/MANUAL/TOKEN/CHIP/TAP/RFID. See 'Card Entry Options' in Chapter 5 for more detail.

Id	N	String	1-20	Used by the POS system to track the transaction e.g. invoice # or receipt #. Sent unchanged in response.

Table 60

Example Response

A successful response:

```
<PLResponse>
    <Result>APPROVED</Result>
    <Command>EBTVOID</Command>
    <ResultText>Approved</ResultText>
    <RecNum>3795</RecNum>
    <MerchantId>000012102012</MerchantId>
    <TerminalId>002</TerminalId>
    <Id>9999</Id>
    <InputMethod>MANUAL</InputMethod>
</PLResponse>
```

EBT Withdrawal

This transaction is used to withdraw cash benefits for the cardholder.

Request Elements

Element Tag	Required	Type	Size	Notes
Command	Y	Fixed String	-	EBTWITHDRAWAL
RecNum	Y	String	4	Reference to the original transaction to be found.
Amount	Y	Float	1-7	The original transaction amount
Input	Y	Fixed String	-	Specifies type of card data given. Must be one of : SWIPED/MANUAL/EXTERNAL/TOKEN
Id	N	String	1-20	Used by the POS system to track the transaction e.g. invoice # or receipt #. Sent unchanged in response.

Table 80

Example Request

A manual transaction:

```
<Input>MANUAL</Input>
    <Command>EBTWITHDRAWAL</Command>
    <CardNumber>9999999812000005</CardNumber>
    <Amount>5.00</Amount>
    <Id>9999</Id>
</PLRequest>
```

Example Response

A successful response:

```
<PLResponse>
    <Result>APPROVED</Result>
    <Command>EBTWITHDRAWAL</Command>
    <ResultText>Approved</ResultText>
    <RecNum>3795</RecNum>
    <MerchantId>000012102012</MerchantId>
    <TerminalId>002</TerminalId>
    <Amount>5.00</Amount>
    <InputMethod>MANUAL</InputMethod>
    <Id>9999</Id>
</PLResponse>
```

Pre-Paid Credit/Debit Cards

Some pre-paid cards may be used in a similar manner to credit or debit cards, but are loaded in the same manner as a gift card. Money is loaded on the card in advance and subsequently, can be used in the same manner as a credit card.

Currently, these cards are supported through other existing commands (subject to change as functionality is required).

The following commands can be used with Pre-Paid Credit/Debit cards:

CCSALE	Process a sale transaction, reducing the balance on a card.
GCACTIVATE	Activates a card and add a starting balance to the card.
GCRELOAD	Adds funds to the card.
GCBALANCE	Determines balance on the card.
GCVOID / CCVOID	Voids a previous transaction. e.g. CCSALE or GCRELOAD

Loyalty Transactions

Posting Transactions

All transactions may be posted to:

https://giftserver.precidia.com/payments/http_reward.php

Data is sent in XML format, tied to variable RawTranData.

An example of the XML format is:

```
RawTranData=<PGRequest>
                    <Command>GetReward</Command>
                    <Program>6</Program>
                    <TID>14</TID>
              </PGRequest>
```

Socket Transactions (Cancelled)

Transactions may be sent over an SSL socket to https://giftserver.precidia.com:8080

Transactions should be in XML format.

An example of the XML format is:

```
<PGRequest>
<Command>GetReward</Command>
<Program>6</Program>
<TID>14</TID>
</PGRequest>
```

Development

Transactions may be sent over a TCP socket (non-SSL) to https://giftserver.precidia.com:9000

Transactions should be in XML format.

An example of the XML format is:

NOTE: ** CURRENTLY DISABLED **

```
<PGRequest>
<Command>GetReward</Command>
<Program>6</Program>
<TID>14</TID>
</PGRequest>
```

Transaction Behavior/Recovery

In the event of a failed transaction due to connections breaking and similar issues, reconnect to the server and resend the message.

Request a New Reward Code

Get a new reward code from the server for the specified program. This will generate a new reward code in the selected program and return it to the terminal. This can be used to generate any static code – it cannot generate a card-linked code.

Request Elements

Element Tag	Required	Type	Size	Notes
Command	Y	Fixed String	-	GetReward
Program	Y	Int	1-10	Specifies the program to fetch a code for
Id	N	String	1-20	Id used by the POS system to track the transaction e.g. invoice # or receipt #.
TID	Y	String	1-20	Terminal ID of the store

Table 81

Example Request:

```
<PGRequest>
    <Command>GetReward</Command>
    <Program>6</Program>
    <Id>9999</Id>
    <TID>14</TID>
</PGRequest>
```

Response Elements

Element Tag	Required	Type	Size	Notes
Command	Y	Fixed String	-	GetReward
RewardCode	Y	Int	1-10	The reward code requested. Present for approved transactions only.
Id	N	String	1-20	Id used by the POS system to track the transaction e.g. invoice # or receipt #.
Result	Y	Fixed String	-	Success/Error
ErrorCode	Y	Int	4	0000 for success
ResultText	Y	String	1-20	Text to describe result/explain error

Table 82

Example Response:

```
<PGResponse>
    <Command>GetReward</Command>
    <Result>Success</Result>
    <ResultText>Success</ResultText>
    <ErrorCode>0000</ErrorCode>
    <RewardCode>683485</RewardCode>
    <Id>9999</Id>
</PGResponse>
```

Request the Status of a Reward Code

This command checks the status of a reward code to see if it can be used. It returns whether or not the reward code is valid at this time.

Returns either a 1 or 0, and the result may vary depending on the date.

> **NOTE:** This is a simple query function and will not change any transaction information.

Request Elements

Element Tag	Required	Type	Size	Notes
Command	Y	Fixed String	-	Status
RewardCode	Y	Int	6-16	The code to validate
Id	N	String	1-20	Id used by the POS system to track the transaction e.g. invoice # or receipt #.
TID	Y	String	1-20	Terminal ID of the store
UTC	Y	String	16	Current local date format: DDMMYYYY HH:MM:SS format

Table 83

Example Request

```
<PGRequest>
      <Command>Status</Command>
      <RewardCode>689543</RewardCode>
      <Id>9999</Id>
      <TID>14</TID>
      <UTC>10072012 11:02:29</UTC>
</PGRequest>
```

Response Elements

Element Tag	Required	Type	Size	Notes
Command	Y	Fixed String	-	Status
RewardStatus	Y	Int	1	1 for active, 0 for inactive
Id	N	String	1-20	Id used by the POS system to track the transaction e.g. invoice # or receipt #.
Result	Y	Fixed String	-	Success/Error
ErrorCode	Y	String	4	0000 for success, otherwise error code
ResultText	Y	String	1-20	Text to describe result/explain error

Table 84

Example Response:

```
<PGResponse>
      <Command>Status</Command>
      <Result>Success</Result>
      <ResultText>Success</ResultText>
      <ErrorCode>0000</ErrorCode>
      <RewardStatus>1</RewardStatus>
      <Id>9999</Id>
</PGResponse>
```

Redeem a Reward Code

This is a two-step transaction which may be repeated over multiple reward codes on a single sale. Reward codes which cannot be used in conjunction with previous rewards are rejected. The transaction applies a reward code, with different results depending on the program. Results may be strictly server-side (incrementing a stored value, changing a gift card balance), or result in changes to the transaction (price reduction to an item, a discount to the total value, etc.) Most times, responses will include a message to be applied to the receipt.

Element Tag	Required	Type	Size	Notes
Command	Y	Fixed String	-	REDEEM
RewardCode	Y	Int	6-16	The code to validate
Id	N	String	1-20	Id used by the POS system to track the transaction e.g. invoice # or receipt #.
TID	Y	String	1-20	Terminal ID of the store

Table 85

Example Request

```
<PGRequest>
	<Command>REDEEM</Command>
	<RewardCode>864576</RewardCode>
	<Id>9999</Id>
	<TID>14</TID>
</PGRequest>
```

The initial response will contain blank fields. These values must be sent back to the server for the transaction to be processed. In the event a request is sent with missing data, the response will be returned with the empty fields until all data is sent up.

Element Tag	Req	Type	Size	Notes
Command	Y	Fixed String	-	REDEEM
RewardCode	Y	Int		The code to validate
Id	N	String	1-20	Id used by the POS system to track the transaction e.g. invoice # or receipt #.
Result	Y	Fixed String	-	Success/Data/Error
ErrorCode	Y	Int	4	0000 for success/data, otherwise error code
UTC	N	String		Current local date in DDMMYYYY HH:MM:SS format
Total	N	Float	3-7	Total dollar amount for the transaction
GiftCard	N	Int		Gift card number associated with the reward program
CRM	N	Int		Authorization ID for CRM-linked codes
ProgramsApplid	N	String		0, or previously sent ProgramsApplied values, ; separated
PLUs	N	XML		PLU/price information in XML format

Table 86

Example Response

```
<PGResponse>
	<Command>REDEEM</Command>
	<Result>Data</Result>
	<ResultText>Data</ResultText>
	<ErrorCode>0000</ErrorCode>
	<RewardCode>689543</RewardCode>
	<Id>9999</Id>
	<TID>14</TID>
	<UTC></UTC>
	<PLUs></PLUs>
	<Total></Total>
	<GiftCard></GiftCard>
	<ProgramsApplied></ProgramApplied>
</PGResponse>
```

The second stage involves a request sent back up, with all fields filled in. All fields present in the first response are required, all fields which were not present in the first response are not required.

Element Tag	Req	Type	Size	Notes
Command	Y	Fixed String	-	REDEEM
RewardCode	Y	Int		The code to validate
Id	N	String	1-20	Id used by the POS system to track the transaction e.g. invoice # or receipt #.
UTC	N	String		Current local date in UTC format
Total	N	Float	3-7	Total dollar amount for the transaction
GiftCard	N	Int		Gift card number associated with the reward program
CRM	N	Int		Authorization ID for CRM-linked cards
ProgramsApplied	N	String		0, or previously sent ProgramsApplied values, ; separated
PLUs	N	XML		PLU/price information in XML format

Table 87

Example Request

```
<PGRequest>
      <Command>REDEEM</Command>
      <RewardCode>689543</RewardCode>
      <Id>9999</Id>
      <TID>14</TID>
      <UTC>10072012 11:02:29</UTC>
      <PLUs>
            <Item><PLU>COFF01</PLU><Price>1.00</Price></Item>
            <Item><PLU>DON02</PLU><Price>0.50</Price></Item>
            <Item><PLU>COFF03</PLU><Price>2.25</Price></Item>
      </PLUs>
      <Total>3.75</Total>
      <GiftCard>8765234546601001</GiftCard>
      <ProgramsApplied>215;555</ProgramApplied>
</PGRequest>
```

Element Tag	Req	Type	Size	Notes
Command	Y	Fixed String	-	Redeem
Id	N	String	1-20	Id used by the POS system to track the transaction e.g. invoice # or receipt #.
Result	Y	Fixed String	-	Success/Error
ErrorCode	Y	Int	4	0000 for success, otherwise error code
ResultText	Y	String	1-20	Text to describe result/explain error
Authcode	Y	Int	1-10	Authorization number
ProgramsApplied	Y	String	1-20	Updated list of all programs applied to the transaction, ; separated
PLUs	N	XML		Any new PLUs to add – typically single reward PLU
Message	N	String	1-20	Text to be applied to the receipt
GiftCard	N	Int	1-10	Gift Card number associated with the reward code
GiftCardBalance	N	Float	3-10	Updated gift card balance, if the reward code modified it

Table 88

Example Response

```
<PGResponse>
      <Command>REDEEM</Command>
      <Id>9999</Id>
      <Result>Success</Result>
      <ResultText>Success</ResultText>
      <ErrorCode>0000</ErrorCode>
      <AuthCode>123456</Authcode>
      <TID>XXX</TID>
      <UTC>10072012 11:02:29</UTC>
      <PLUs>
            <Item><PLU>Reward689543</PLU><Price>-1.00</Price></Item>
      </PLUs>
      <GiftCard>8765234546601001</GiftCard>
      <ProgramsApplied>215;555;867</ProgramApplied>
      <GiftCardBalance>25.35</GiftCardBalance>
      <Message>Enjoy your free coffee!</Message>
</PGResponse>
```

Cancel a Redemption

When a reward code is redeemed, it is queued but not finalized. This means that several aspects of the redemption may not have yet occurred, and that the redemption can be canceled. When a reward code is canceled, the reward code can be used again freely. This will not reset the transaction.

Element Tag	Req	Type	Size	Notes
Command	Y	Fixed String	-	Cancel
AuthCode	Y	Int		Auth code to cancel
Id	N	String	1-20	Id used by the POS system to track the transaction e.g. invoice # or receipt #.
TID	Y	String	1-20	Terminal ID of the store

Table 89

Example Request:

```
<PGRequest>
      <Command>Cancel</Command>
      <AuthCode>864576</RewardCode>
      <Id>9999</Id>
      <TID>14</TID>
</PGRequest>
```

Element Tag	Req	Type	Size	Notes
Command	Y	Fixed String	-	Cancel

Id	N	String	1-20	Id used by the POS system to track the transaction e.g. invoice # or receipt #.
Result	Y	Fixed String	-	Success/Error
ErrorCode	Y	Int	4	0000 for success, otherwise error code
ResultText	Y	String	1-20	Text to describe result/explain error
AuthCode	Y	Int	1-10	Authorization number
RewardCode	Y	Int	1-10	Reward Code associated with the authorization

Table 90

Example Response:

```
<PGResponse>
      <Command>Cancel</Command>
      <Id>9999</Id>
      <Result>Success</Result>
      <ResultText>Success</ResultText>
      <ErrorCode>0000</ErrorCode>
      <AuthCode>123456</Authcode>
</PGResponse>
```

Finalize a Redemption

When a reward code is redeemed, it is queued but not finalized. This means that several aspects of the redemption may not have yet occurred, and that the redemption can be canceled. When the payment transaction has been approved, the finalize transaction should be sent up to the gift server. Once this occurs, the queued transactions will be flagged as finished, and will no longer be able to be canceled.

Element Tag	Req	Type	Size	Notes
Command	Y	Fixed String	-	Finalize
AuthCode	Y	Int	1-10	Auth codes to finalize, ';' separated
Id	N	String	1-20	Id used by the POS system to track the transaction e.g. invoice # or receipt #.
TID	Y	String	1-20	Terminal ID of the store

Table 91

Example Request:

```
<PGRequest>
      <Command>Finalize</Command>
      <AuthCode>864576;679534</RewardCode>
      <Id>9999</Id>
      <TID>14</TID>
</PGRequest>
```

Element Tag	Req	Type	Size	Notes

Command	Y	Fixed String	-	Finalized
Id	N	String	1-20	Id used by the POS system to track the transaction e.g. invoice # or receipt #.
Result	Y	Fixed String	-	Success/Error
ErrorCode	Y	Int	4	0000 for success, otherwise error code
ResultText	Y	String	1-20	Text to describe result/explain error
Authcode	Y	Int	1-10	Authorization numbers finalized, ';' separated

Table 92

Example Response:

```
<PGResponse>
      <Command>Finalized</Command>
      <Id>9999</Id>
      <Result>Success</Result>
      <ResultText>Success</ResultText>
      <ErrorCode>0000</ErrorCode>
      <AuthCode>123456;679534</Authcode>
</PGResponse>
```

Void a Redemption

A reward code that has been applied can be voided. This will re-enable the reward code for use elsewhere. Voiding a reward code used will not reset the transaction. This transaction is intended only to re-enable the code in event of an error. In the special case of voiding a code that was generated with a transaction, the code will be deleted from the server.

Element Tag	Req	Type	Size	Notes
Command	Y	Fixed String	-	RewardVoid
RewardCode	Y	Int	6-16	The code to void
AuthCode	Y	Int	1-10	The authorization code from the redemption
Id	N	String	1-20	Id used by the POS system to track the transaction e.g. invoice # or receipt #.
TID	Y	String	1-20	Terminal ID of the store

Table 93

Example Request:

```
<PGRequest>
      <Command>RewardVoid</Command>
      <RewardCode>689543</RewardCode>
      <AuthCode>123456</AuthCode>
      <Id>9999</Id>
      <TID>14</TID>
</PGRequest>
```

Element Tag	Req	Type	Size	Notes
Command	Y	Fixed String	-	RewardVoid
Id	N	String	1-20	Id used by the POS system to track the transaction e.g. invoice # or receipt #.
Result	Y	Fixed String	-	Success/Error
ErrorCode	Y	String	4	0000 for success, otherwise error code
Authcode	Y	Int	1-10	Authorization code for the void
ResultText	Y	String	1-20	Text to describe result/explain error

Table 94

Example Response:

```
<PGResponse>
      <Command>RewardVoid</Command>
      <Id>9999</Id>
      <Result>Success</Result>
      <ResultText>Success</ResultText>
      <ErrorCode>0000</ErrorCode>
      <Authcode>234567</AuthCode>
</PGResponse>
```

Error Codes

Error	Definition
0000	Success
0001	System Error
1000	Authentication Invalid (CRM)
2000	Program not active (wrong day)
2001	Program not active (Expired)
2002	Program not yet started
3000	Invalid Data
4000	Conditions not met (buy 1 get 1 free..only bought 1)
4001	Program dependency unmet
5000	Program conflict (two programs that cannot be used on the same transaction)
6000	Card not linked to program (card linked program, with card provided not part of program)
7000	No such reward code
7001	Auth code invalid
7002	Auth code expired
8000	Program cannot be voided
9000	No codes available
9999	Program does not exist

Table 95

Administrative (Batch) Transactions

Batch Summary

This transaction provides a summary of information that is in the current open batch. Detailed data is returned in XML data fields and also as a report in the *<Receipt>* tags, which can be printed to a standard 42 character receipt printer.

> **NOTE:** A printer device must be defined (and assigned to the lane) via the POSLynx configuration GUI.

Element Tag	Required	Type	Size	Notes
Command	Y	Fixed String	-	BATCHSUMMARY
BatchIndex	N	String		If provided it will allow to get summary for a single batch.

Table 96

Example Request

```
<PLRequest>
    <Command>BATCHSUMMARY</Command>
    <BatchIndex>123</BatchIndex>
</PLRequest>
```

Element Tag	Req	Type	Size	Notes
Command	Y	Fixed String	-	BATCHSUMMARY
TotalItemCount	Y	Integer	1-5	Total number of items in current open batch.
TotalAmount	Y	Float	3-7	Total dollar amount of current open batch.
CCSaleCount	N	Integer	1-5	Number of credit card sale transactions in current batch.
CCSaleAmount	N	Float	3-7	Total dollar amount of credit card sales in current batch.
CCRefundCount	N	Integer	1-5	Number of credit card refund transactions in current batch.
CCRefundAmount	N	Float	3-7	Total dollar amount of credit card refunds in current batch.
DCSaleCount	N	Integer	1-5	Number of debit card sale transactions in current batch.
DCSaleAmount	N	Float	3-7	Total dollar amount of debit card sales in

				current batch.
DCRefundCount	N	Integer	1-5	Number of debit card refund transactions in current batch.
DCRefundAmount	N	Float	3-7	Total dollar amount of debit card refunds in current batch.
GCSaleCount	N	Integer	1-5	Number of gift card sale transactions in current batch.
GCSaleAmount	N	Float	3-7	Total dollar amount of gift card sales in current batch.
GCRefundCount	N	Integer	1-5	Number of gift card refund transactions in current batch.
GCRefundAmount	N	Float	3-7	Total dollar amount of gift card refunds in current batch.
GCActivateCount	N	Integer	1-5	Number of gift card activation transactions in current batch.
GCActivateAmount	N	Float	3-7	Total dollar amount of gift card activations in current batch.
GCVoidCount	N	Integer	1-5	Number of gift card void transactions in current batch.
GCVoidAmount	N	Float	3-7	Total dollar amount of gift card voids in current batch.
GCSaleCountHost	N	Integer	1-5	Number of gift card sale transactions in current host batch.
GCSaleAmountHost	N	Float	3-7	Total dollar amount of gift card sales in current host batch.
GCRefundCountHost	N	Integer	1-5	Number of gift card refund transactions in current host batch.
GCRefundAmountHost	N	Float	3-7	Total dollar amount of gift card refunds in current host batch.
GCActivateCountHost	N	Integer	1-5	Number of gift card activation transactions in current host batch.
GCActivateAmountHost	N	Float	3-7	Total dollar amount of gift card activations in current host batch.
GCVoidCountHost	N	Integer	1-5	Number of gift card void transactions in current host batch.
GCVoidAmountHost	N	Float	3-7	Total dollar amount of gift card voids in current host batch.
<Receipt>	N			Contains a formatted line by line receipt for simple printing.

Table 97

NOTE: Not all Gift hosts support the above tabulated data being returned. The host data can be compared to the local POSLynx data and some discrepancies may exist. Corrections can be made before closing the batch. If the batch is already closed, the host's data will be used.

Example Response

```
<PLResponse>
    <Result>APPROVED</Result>
    <Command>BATCHSUMMARY </Command>
    <TotalItemCount>13</TotalItemCount>
    <TotalAmount>123.00</TotalAmount>
    <CCSaleCount>6</CCSaleCount>
    <CCSaleAmount>100.00</CCSaleAmount>
    <CCRefundCount>0</CCRefundCount>
    <CCRefundAmount>0.00</CCRefundAmount>
    <DCSaleCount>3</DCSaleCount>
    <DCSaleAmount>30.00</DCSaleAmount>
    <DCRefundCount>1</DCRefundCount>
    <DCRefundAmount>1.00</DCRefundAmount>
    <GCSaleCount>1</GCSaleCount>
    <GCSaleAmount>5.00</GCSaleAmount>
    <GCRefundCount>1</GCRefundCount>
    <GCRefundAmount>1.00</GCRefundAmount>
    <GCActivateCount>1</GCActivateCount>
    <GCActivateAmount>10.00</GCActivateAmount>
    <GCVoidCount>0</GCVoidCount>
    <GCVoidAmount>0.00</GCVoidAmount>
    <GCSaleCountHost>1</GCSaleCountHost>
    <GCSaleAmountHost>5.00</GCSaleAmountHost>
    <GCRefundCountHost>1</GCRefundCountHost>
    <GCRefundAmountHost>1.00</GCRefundAmountHost>
    <GCActivateCountHost>1</GCActivateCountHost>
    <GCActivateAmountHost>10.00</GCActivateAmountHost>
    <GCVoidCountHost>0</GCVoidCountHost>
    <GCVoidAmountHost>0.00</GCVoidAmountHost>
    <ErrorCode>0000</ErrorCode>
    <Receipt>
        <Receipt1>Batch Summary</Receipt1>
        <Receipt2>----------------------------------------</Receipt2>
        <Receipt3 />
        <Receipt4>Batch item count:                    13</Receipt4>
        <Receipt5>Net batch total:                 123.00</Receipt5>
        <Receipt6>Credit purchase count:                6</Receipt6>
```

```
        <Receipt7>Credit purchase amount:          100.00</Receipt7>
        <Receipt8>Credit refund count:                  0</Receipt8>
        <Receipt9>Credit refund amount:              0.00</Receipt9>
        <Receipt10>Debit purchase count:                3</Receipt10>
        <Receipt11>Debit purchase amount:            30.00</Receipt11>
        <Receipt12>Debit refund count:                  1</Receipt12>
        <Receipt13>Debit refund amount:              1.00</Receipt13>
        <Receipt14>Gift Sale Count:                     1</Receipt14>
        <Receipt15>Gift Sale Amount:                 5.00</Receipt15>
        <Receipt16>Gift Refund Count:                   1</Receipt16>
        <Receipt17>Gift Refund Amount:               1.00</Receipt17>
        <Receipt18>Gift Activate Count:                 1</Receipt18>
        <Receipt19>Gift Activate Amount:            10.00</Receipt19>
        <Receipt20>Gift Void Count:                     0</Receipt20>
        <Receipt21>Gift Void Amount:                 0.00</Receipt21>
        <Receipt14>Host Gift Sale Count:                1</Receipt14>
        <Receipt15>Host Gift Sale Amount:            5.00</Receipt15>
        <Receipt16>Host Gift Refund Count:              1</Receipt16>
        <Receipt17>Host Gift Refund Amount:          1.00</Receipt17>
        <Receipt18>Host Gift Activate Count:            1</Receipt18>
        <Receipt19>Host Gift Activate Amount:       10.00</Receipt19>
        <Receipt20>Host Gift Void Count:                0</Receipt20>
        <Receipt21>Host Gift Void Amount:            0.00</Receipt21>
    </Receipt>
</PLResponse>
```

Group Batch Summary

This transaction provides a summary of information that is in the current open batch on multiple lanes. Depending on the configuration in the "Batch management" settings, this command can trigger individual BATCHSUMMARY commands to be sent, to up to 15 lanes, in sequence with 10 minutes time out for each command (during which the TNP-CG may appear unresponsive for long durations).

Detailed data is returned in XML data fields, with each individual lane response being shown with its lane name, IP address and port.

After all individual responses are listed, a summary for all lanes will be added to the main response.

Element Tag	Required	Type	Size	Notes
Command	Y	Fixed String	-	GROUPBATCHSUMMARY

Table 98

Example Request

```
<PLRequest>
    <Command>GROUPBATCHSUMMARY</Command>
</PLRequest>
```

Element Tag	Required	Type	Size	Notes
Command	Y	Fixed String	-	GROUPBATCHSUMMARY
LANEBATCHSUMMARY	y			The response for a BATCHSUMMARY command for a given lane, with the Lane name, IP and Port added into it.
GroupTotalItemCount	Y	Integer	1-5	Total number of items in all batches in the group.
GroupTotalAmount	Y	Float	3-7	Total dollar amount of all batches in the group.
GroupCCSaleCount	N	Integer	1-5	Number of credit card sale transactions in all batches in the group.
GroupCCSaleAmount	N	Float	3-7	Total dollar amount of credit card sales in all batches in the group.
GroupCCRefundCount	N	Integer	1-5	Number of credit card refund transactions in all batches in the group.
GroupCCRefundAmount	N	Float	3-7	Total dollar amount of credit card refunds in all batches in the group.
GroupDCSaleCount	N	Integer	1-5	Number of debit card sale transactions in all batches in the group.
GroupDCSaleAmount	N	Float	3-7	Total dollar amount of debit card sales in all batches in the group.

GroupDCRefundCount	N	Integer	1-5	Number of debit card refund transactions in all batches in the group.
GroupDCRefundAmount	N	Float	3-7	Total dollar amount of debit card refunds in all batches in the group.
GroupGCSaleCount	N	Integer	1-5	Number of gift card sale transactions in all batches in the group.
GroupGCSaleAmount	N	Float	3-7	Total dollar amount of gift card sales in all batches in the group.
GroupGCRefundCount	N	Integer	1-5	Number of gift card refund transactions in all batches in the group.
GroupGCRefundAmount	N	Float	3-7	Total dollar amount of gift card refunds in all batches in the group.
GroupGCActivateCount	N	Integer	1-5	Number of gift card activation transactions in all batches in the group.
GroupGCActivateAmount	N	Float	3-7	Total dollar amount of gift card activations in all batches in the group.
GroupGCVoidCount	N	Integer	1-5	Number of gift card void transactions in all batches in the group.
GroupGCVoidAmount	N	Float	3-7	Total dollar amount of gift card voids in all batches in the group.
GroupGCSaleCountHost	N	Integer	1-5	Number of gift card sale transactions in all batches in the group.
GroupGCSaleAmountHost	N	Float	3-7	Total dollar amount of gift card sales in all batches in the group.
GroupGCRefundCountHost	N	Integer	1-5	Number of gift card refund transactions in all batches in the group.
GroupGCRefundAmountHost	N	Float	3-7	Total dollar amount of gift card refunds in all batches in the group.
GroupGCActivateCountHost	N	Integer	1-5	Number of gift card activation transactions in all batches in the group.
GroupGCActivateAmountHost	N	Float	3-7	Total dollar amount of gift card activations in all batches in the group.
GroupGCVoidCountHost	N	Integer	1-5	Number of gift card void transactions in current host batch.
GroupGCVoidAmountHost	N	Float	3-7	Total dollar amount of gift card voids in current host batch.

Table 99

Example Response

```
<PLResponse>
            <Result>APPROVED</Result>
            <Command>GROUPBATCHSUMMARY</Command>
      <LANEBATCHSUMMARY>
          <Result>APPROVED</Result>
          <LaneName>StoreFront</LaneName>
          <LaneIPAddress>10.10.10.10</LaneIPAddress>
```

```
            <LanePort>9300</LanePort>
                        :
                        :
                        :
                    SAME AS BATCHSUMMARY
                        :
                        :
                        :
        </LANEBATCHSUMMARY>
        <LANEBATCHSUMMARY>
            <Result>APPROVED</Result>
            <LaneName>BackOffice</LaneName>
            <LaneIPAddress>10.10.10.11</LaneIPAddress>
            <LanePort>9400</LanePort>
                        :
                        :
                        :
                    SAME AS BATCHSUMMARY
                        :
                        :
                        :
        </LANEBATCHSUMMARY>
              <GroupTotalItemCount>7</GroupTotalItemCount>
        <GroupTotalAmount>112.00</GroupTotalAmount>
              <GroupCCSaleCount>4</GroupCCSaleCount>
              <GroupCCSaleAmount>110.00</GroupCCSaleAmount>
              <GroupCCRefundCount>1</GroupCCRefundCount>
              <GroupCCRefundAmount>10.00</GroupCCRefundAmount>
              <GroupDCSaleCount>1</GroupDCSaleCount>
              <GroupDCSaleAmount>15.00</GroupDCSaleAmount>
              <GroupDCRefundCount>1</GroupDCRefundCount>
        <GroupDCRefundAmount>3.00</GroupDCRefundAmount>
<PLResponse>
```

Batch Close

This transaction closes a batch on the host processor. It will Transfer all funds from the processor to the merchant. Detailed data is returned in XML data fields and also as a report in the *<Receipt>* tags, which can be printed to a standard 42 character Receipt Printer.

Element Tag	Required	Type	Size	Notes
Command	Y	Fixed String	-	BATCHCLOSE
IPAddress	N	String	7-15	Override the configured POSlynx IP Address. Allows controlling which POSLynx processes each transaction individually. Only applicable to TNP-CG.
IPPort	N	Numeric	4-5	Override the configured POSLynx port. Used with <IPAddress> to control which POSLynx and port are used to process each transaction. Only applicable to TNP-CG.

Table 100

Example Request

```
<PLRequest>
    <Command>BATCHCLOSE</Command>
</PLRequest>
```

Element Tag	Required	Type	Size	Notes
Command	Y	Fixed String	-	BATCHCLOSE
Result	Y	Fixed String	-	Result of all included batches – will be APPROVED, DECLINED, ERROR. I.e will only be APPROVED if all batches are approved.
TotalItemCount	Y	Integer	1-5	Total number of items in all closed batches
TotalAmount	Y	Float	3-7	Total dollar amount of all closed batches
Batch	Y	Container		Container for individual batch details
Host	Y	String	1-20	Name of host for individual batch
BatchResult	Y	Fixed String	-	APPROVED, DECLINED, ERROR
BatchResultText	Y	String	1-20	Text describing result – contains the error message from host if error occurred
BatchItemCount	Y	Integer	1-10	Count of items in the individual batch
BatchTotal	Y	Float	3-7	Total dollar amount of the individual batch
BatchNumber	Y	Integer	1-10	Batch number of batch on host
BatchErrorCode	Y	Integer	1-10	Result of batch close. 0000 if batch

				closed properly
CCSaleCount	N	Integer	1-10	Number of Credit Card items in current batch.
CCSaleAmount	N	Float	3-7	Total dollar amount of Credit sales in current batch.
CCRefundCount	N	Integer	1-5	Number of Credit Card items returned in current batch.
CCRefundAmount	N	Float	3-7	Total dollar amount of Credit returns in current batch.
DCSaleCount	N	Integer	1-5	Number of Debit Card items in current batch.
DCSaleAmount	N	Float	3-7	Total dollar amount of Debit sales in current batch.
DCRefundCount	N	Integer	1-5	Number of Debit Card items returned in current batch.
DCRefundAmount	N	Float	3-7	Total dollar amount of Debit returns in current batch.
MerchantId	N	String	1-20	Merchant Id used to process transaction with host.
TerminalId	N	String	1-20	TerminalId used to process transaction with host.
<Receipt>	N			Contains a formatted line by line receipt for simple printing.

Table 101

Often there will only be a single batch, however it is possible to have more than one host configured and therefore there will be multiple batches. Each batch has details reported in a separate <*Batch*> section of the Resultant XML.

Example Response

```
<PLResponse>
      <Command>BATCHCLOSE</Command>
      <Result>ERROR</Result>
      <TotalItemCount>7</TotalItemCount>
      <TotalAmount>112.00</TotalAmount>
       <CCSaleCount>4</CCSaleCount>
       <CCSaleAmount>110.00</CCSaleAmount>
       <CCRefundCount>1</CCRefundCount>
       <CCRefundAmount>10.00</CCRefundAmount>
       <DCSaleCount>1</DCSaleCount>
       <DCSaleAmount>15.00</DCSaleAmount>
       <DCRefundCount>1</DCRefundCount>
      <DCRefundAmount>3.00</DCRefundAmount>
      <MerchantId>700000000373</MerchantId>
```

```
      <TerminalId>001</TerminalId>
<Batch>
            <Host>Hostname1</Host>
            <BatchResult>APPROVED</Result>
                  <BatchResultText>Batch Closed Successfully</ResultText>
                        <BatchItemCount>2</ItemCount>
            <BatchTotal>12.00</BatchTotal>
            <BatchNumber>1</BatchNumber>
            <BatchErrorCode>0000</BatchErrorCode>
      </Batch>
      <Batch>
            <Host>Hostname2</Host>
            <BatchResult>APPROVED</Result>
                  <BatchResultText>Batch Closed Successfully</ResultText>
                        <BatchItemCount>5</ItemCount>
            <BatchTotal>100.00</BatchTotal>
            <BatchNumber>43</BatchNumber>
            <BatchErrorCode>0000</BatchErrorCode>
      </Batch>
      <Batch>
            <Host>Hostname3</Host>
            <BatchResult>ERROR</Result>
                  <ResultText>Duplicate Batch Number</ResultText>
            <BatchErrorCode>0092</BatchErrorCode>
      </Batch>
      <Receipt>
            <Receipt1>Batch Close Report</Receipt1>
            <Receipt2>----------------------------------------</Receipt2>
                  <Receipt3 />
            <Receipt4>Host: Hostname1</Receipt4>
            <Receipt5>Batch Closed</Receipt5>
            <Receipt6>Batch Number                          1</Receipt6>
            <Receipt7>Batch item count:                     2</Receipt7>
            <Receipt8>Batch Total:                      12.00</Receipt8>
            <Receipt9>----------------------------------------</Receipt9>
            <Receipt10 />
            <Receipt11>Host: Hostname2</Receipt11>
            <Receipt12>Batch Closed</Receipt12>
            <Receipt13>Batch Number                        43</Receipt13>
            <Receipt14>Batch item count:                    5</Receipt14>
            <Receipt15>Batch Total:                    100.00</Receipt15>
            <Receipt16>----------------------------------------</Receipt16>
```

```
            <Receipt17 />
            <Receipt18>Host: Hostname3</Receipt11>
            <Receipt19>Batch ERROR</Receipt12>
            <Receipt20>----------------------------------------</Receipt20>
            <Receipt21 />
            <Receipt22>Total item count:                    7</Receipt22>
            <Receipt23>Net batch total :                112.00</Receipt23>
            <Receipt24>Credit purchase count:               4</Receipt24>
            <Receipt25>Credit purchase amount:          110.00</Receipt25>
            <Receipt26>Credit refund count:                 1</Receipt26>
            <Receipt27>Credit refund amount:             10.00</Receipt27>
            <Receipt28>Debit purchase count:                1</Receipt28>
            <Receipt29>Debit purchase amount:            15.00</Receipt29>
            <Receipt30>Debit refund count:                  1</Receipt30>
            <Receipt31>Debit refund amount:               3.00</Receipt31>
            <Receipt32>2 Batches Closed. 1 Batches Failed.</Receipt32>
      </Receipt>
</PLResponse>
```

Group Batch Close

This transaction closes the current open batch on multiple lanes. Depending on the configuration in the "Batch management" settings, this command can trigger individual BATCHCLOSE commands to be sent to up to 15 lanes in sequence with 10 minutes timeout for each command (during which, the TNP-CG may appear unresponsive for long durations).

Detailed data is returned in XML data fields, each individual lane response shown with its lane name, IP address and port.

After all individual responses are listed, a summary for all lanes will be added to the main response.

Element Tag	Required	Type	Size	Notes
Command	Y	Fixed String	-	GROUPBATCHCLOSE

Table 102

Example Request

```
<PLRequest>
    <Command>GROUPBATCHCLOSE</Command>
</PLRequest>
```

Element Tag	Required	Type	Size	Notes

Command	Y	Fixed String	-	GROUPBATCHCLOSE
LANEBATCHCLOSE	y			The response for a GROUPBATCHCLOSE command for a given lane, with the Lane name, IP and Port added into it.
GroupTotalItemCount	Y	Integer	1-10	Total number of items in all batches in the group.
GroupTotalAmount	Y	Float	3-7	Total dollar amount of all batches in the group.
GroupCCSaleCount	N	Integer	1-10	Number of credit card sale transactions in all batches in the group.
GroupCCSaleAmount	N	Float	3-7	Total dollar amount of credit card sales in all batches in the group.
GroupCCRefundCount	N	Integer	1-10	Number of credit card refund transactions in all batches in the group.
GroupCCRefundAmount	N	Float	3-7	Total dollar amount of credit card refunds in all batches in the group.
GroupDCSaleCount	N	Integer	1-10	Number of debit card sale transactions in all batches in the group.
GroupDCSaleAmount	N	Float	3-7	Total dollar amount of debit card sales in all batches in the group.
GroupDCRefundCount	N	Integer	1-10	Number of debit card refund transactions in all batches in the group.
GroupDCRefundAmount	N	Float	3-7	Total dollar amount of debit card refunds in all batches in the group.
GroupGCSaleCount	N	Integer	1-10	Number of gift card sale transactions in all batches in the group.
GroupGCSaleAmount	N	Float	3-7	Total dollar amount of gift card sales in all batches in the group.
GroupGCRefundCount	N	Integer	1-10	Number of gift card refund transactions in all batches in the group.
GroupGCRefundAmount	N	Float	3-7	Total dollar amount of gift card refunds in all batches in the group.
GroupGCActivateCount	N	Integer	1-10	Number of gift card activation transactions in all batches in the group.
GroupGCActivateAmount	N	Float	3-7	Total dollar amount of gift card activations in all batches in the group.
GroupGCVoidCount	N	Integer	1-10	Number of gift card void transactions in all batches in the group.
GroupGCVoidAmount	N	Float	3-7	Total dollar amount of gift card voids in all batches in the group.
GroupGCSaleCountHost	N	Integer	1-10	Number of gift card sale transactions in all batches in the group.

GroupGCSaleAmountHost	N	Float	3-7	Total dollar amount of gift card sales in all batches in the group.
GroupGCRefundCountHost	N	Integer	1-10	Number of gift card refund transactions in all batches in the group.
GroupGCRefundAmountHost	N	Float	3-7	Total dollar amount of gift card refunds in all batches in the group.
GroupGCActivateCountHost	N	Integer	1-10	Number of gift card activation transactions in all batches in the group.
GroupGCActivateAmountHost	N	Float	3-7	Total dollar amount of gift card activations in all batches in the group.
GroupGCVoidCountHost	N	Integer	1-10	Number of gift card void transactions in current host batch.
GroupGCVoidAmountHost	N	Float	3-7	Total dollar amount of gift card voids in current host batch.

Table 103

Example Response

```
<PLResponse>
            <Result>APPROVED</Result>
            <Command>GROUPBATCHCLOSE</Command>
      <LANEBATCHCLOSE>
          <Result>APPROVED</Result>
          <LaneName>StoreFront</LaneName>
          <LaneIPAddress>10.10.10.10</LaneIPAddress>
          <LanePort>9300</LanePort>
                      :
                      :
                      :
                SAME AS BATCHCLOSE
                      :
                      :
                      :
      </LANEBATCHCLOSE>
      <LANEBATCHSUMMARY>
          <Result>APPROVED</Result>
          <LaneName>BackOffice</LaneName>
          <LaneIPAddress>10.10.10.11</LaneIPAddress>
          <LanePort>9400</LanePort>
                      :
                      :
```

```
                    :
                SAME AS BATCHCLOSE
                    :
                    :
                    :
      </LANEBATCHCLOSE>
            <GroupTotalItemCount>7</GroupTotalItemCount>
      <GroupTotalAmount>112.00</GroupTotalAmount>
            <GroupCCSaleCount>4</GroupCCSaleCount>
            <GroupCCSaleAmount>110.00</GroupCCSaleAmount>
            <GroupCCRefundCount>1</GroupCCRefundCount>
            <GroupCCRefundAmount>10.00</GroupCCRefundAmount>
            <GroupDCSaleCount>1</GroupDCSaleCount>
            <GroupDCSaleAmount>15.00</GroupDCSaleAmount>
            <GroupDCRefundCount>1</GroupDCRefundCount>
      <GroupDCRefundAmount>3.00</GroupDCRefundAmount>
<PLResponse>
```

Batch Clear

This transaction clears the host batch. This will destroy all transaction information and could result in lost transactions. This should only be used during testing. Ensure transaction records are backed up (or exist in another format) before initiating this command. Detailed data is returned in XML data fields and also as a report in the *<Receipt>* tags, which can be printed to a standard 42 character Receipt Printer.

BEWARE: This command is dangerous and is rarely required in a production environment.

Element Tag	Required	Type	Size	Notes
Command	Y	Fixed String	-	BATCHCLEAR
IPAddress	N	String	7-15	Override the configured POSLynx IP Address. Allows controlling which POSLynx processes each transaction individually. Only applicable to TNP-CG.
IPPort	N	Numeric	4-5	Override the configured POSLynx port. Used with *<IPAddress>* to control which POSLynx and port are used to process each transaction. Only applicable to TNP-CG.

Table 104

Example Request

```
<PLRequest>
    <Command>BATCHCLEAR</Command>
</PLRequest>
```

Element Tag	Required	Type	Size	Notes
Command	Y	Fixed String	-	BATCHCLEAR
Result	Y	Fixed String	-	Worst result of all included batches – will be APPROVED, DECLINED, ERROR. I.e will only be APPROVED if all batches are approved.
ItemCount	Y	Integer	1-10	Total number of items in all closed batches
Total	Y	Float	3-7	Total dollar amount of all closed batches
Batch	Y	Container		Container for individual batch details
Host	Y	String	1-20	Name of host for individual batch
BatchResult	Y	Fixed String	-	APPROVED, DECLINED, ERROR

BatchResultText	Y	String	1-20	Text describing result – contains the error message from host if error occurred
BatchItemCount	Y	Integer	1-10	Count of items in the individual batch
BatchTotal	Y	Float	3-7	Total dollar amount of the individual batch
BatchNumber	Y	Integer	1-10	Batch number of batch on host
BatchErrorCode	Y	Integer	1-10	Result of batch close. 0000 if batch closed properly.
CCSaleCount	N	Integer	1-10	Number of Credit Card items in current batch.
CCSaleAmount	N	Float	3-7	Total dollar amount of Credit sales in current batch.
CCRefundCount	N	Integer	1-10	Number of Credit Card items returned in current batch.
CCRefundAmount	N	Float	3-7	Total dollar amount of Credit returns in current batch.
DCSaleCount	N	Integer	1-10	Number of Debit Card items in current batch.
DCSaleAmount	N	Float	3-7	Total dollar amount of Debit sales in current batch.
DCRefundCount	N	Integer	1-10	Number of Debit Card items returned in current batch.
DCRefundAmount	N	Float	3-7	Total dollar amount of Debit returns in current batch.
Receipt	N			Contains a formatted line by line receipt for simple printing.

Table 105

Often, only a single batch will exist, although it is possible to have more than one host configured (and therefore, multiple batches). Each batch has details reported in a separate *<Batch>* section of the resulting XML.

The results display details of the contents of the cleared batch. The Total dollar amount and Total transactions are included.

Example Response

```
<PLResponse>
      <Command>BATCHCLEAR</Command>
      <Result>APPROVED</Result>
      <ItemCount>4</ItemCount>
      <Total>4.00</BatchTotal>
      <CCSaleCount>2</CCSaleCount>
      <CCSaleAmount>4.00</CCSaleAmount>
      <CCRefundCount>1</CCRefundCount>
```

```
      <CCRefundAmount>1.00</CCRefundAmount>
      <DCSaleCount>1</DCSaleCount>
      <DCSaleAmount>1.00</DCSaleAmount>
      <DCRefundCount>0</DCRefundCount>
      <DCRefundAmount>0.00</DCRefundAmount>
      <Batch>
            <Host>Hostname1</Host>
            <BatchResult>APPROVED</Result>
                  <BatchResultText>Batch CLEARED</ResultText>
             <BatchItemCount>4</ItemCount>
            <BatchTotal>4.00</BatchTotal>
            <BatchNumber>8</BatchNumber>
            <BatchErrorCode>0000</BatchErrorCode>
      </Batch>
      <Receipt>
                  <Receipt1>Batch Clear Report</Receipt1>
            <Receipt2>----------------------------------------</Receipt2>
            <Receipt3 />
            <Receipt4>Host: Hostname1</Receipt4>
            <Receipt5>Batch CLEARED</Receipt5>
            <Receipt6>Batch Number                          8</Receipt6>
            <Receipt7>Batch item count:                     4</Receipt7>
            <Receipt8>Batch Total:                       4.00</Receipt8>
            <Receipt9>----------------------------------------</Receipt9>
            <Receipt17 />
            <Receipt18>Total item count:                       4</Receipt18>
            <Receipt19>Net batch total :                    4.00</Receipt19>
            <Receipt20>Credit purchase count:                  2</Receipt20>
            <Receipt21>Credit purchase amount:              4.00</Receipt21>
            <Receipt22>Credit refund count:                    1</Receipt22>
            <Receipt23>Credit refund amount:                1.00</Receipt23>
            <Receipt24>Debit purchase count:                   1</Receipt24>
            <Receipt25>Debit purchase amount:               1.00</Receipt25>
            <Receipt26>Debit refund count:                     0</Receipt26>
            <Receipt27>Debit refund amount:                 0.00</Receipt27>
            <Receipt28>1 Batches CLEARED. 0 Batches Failed.</Receipt28>
      </Receipt
</PLResponse>
```

Change Batch Number

This transaction changes the current batch number on the host. This command is required sometimes to synchronize with certain hosts.

Element Tag	Required	Type	Size	Notes
Command	Y	Fixed String	-	BATCHCHANGENUM
BatchNum	Y	Integer	1-4	New Batch Number.
HostName	N	String	1-20	This element is needed when a lane contains multiple hosts and a user wants to change the batch number for one of the configured hosts. If this tag is not provided, the batch number will be changed for ALL hosts. ***Note:** this element tag is only supported for the following hosts: • MerchantLink • Paymentech • FD Global • Global Payments
IPAddress	N	String	7-15	Override the configured POSLynx IP Address. Allows controlling which POSLynx processes each transaction individually. Only applicable to TNP-CG.
IPPort	N	Numeric	4-5	Override the configured POSLynx port. Used with *<IPAddress>* to control which POSLynx and port are used to process each transaction. Only applicable to TNP-CG.

Table 106

Example Request

```
<PLRequest>
    <Command>BATCHCHANGENUM</Command>
             <BatchNum>10</BatchNum>
             <BatchIndex>0</BatchIndex>
</PLRequest>
```

Example Response

```
<PLResponse>
    <Result>APPROVED</Result>
    <Command>BATCHCHANGENUM</Command>
    <ResultText>New Batch Number is:10</ResultText>
```

```
    <BatchNum>10</BatchNum>
    <ErrorCode>0000</ErrorCode>
</PLResponse>
```

Item Detail

This batch command obtains information on a previously processed transaction. Detailed data is returned in XML data fields. Also, the information can be obtained as a report in the *<Receipt>* tags, which can be printed to a standard 42 character Receipt Printer.

Element Tag	Required	Type	Size	Notes
Command	Y	Fixed String	-	ITEMDETAIL
RecNum	Y	String	4	Reference to the original transaction to be found.
IPAddress	N	String	7-15	Override the configured POSLynx IP Address. Allows controlling which POSLynx processes each transaction individually. Only applicable to TNP-CG.
IPPort	N	Numeric	4-5	Override the configured POSLynx port. Used with *<IPAddress>* to control which POSLynx and port are used to process each transaction. Only applicable to TNP-CG.

Table 107

Example Request

```
<PLRequest>
    <Command>ITEMDETAIL</Command>
    <RecNum>1101</RecNum>
</PLRequest>
```

Element Tag	Required	Type	Size	Notes
Command	Y	Fixed String	-	ITEMDETAIL
Result	Y	Fixed String	-	APPROVED/ERROR/DECLINED- result of this request. Will be declined if RecNum cannot be found.
RecNum	Y	String	4	Record number of transaction returned (same as input)
ResultText	Y	String	1-20	Text to describe result of this request – e.g. explain error or result that could be placed on POS screen.
ErrorCode	Y	String	4	Error code - 0000 if everything is successful. 1 if Result is DECLINED. Numeric error code if the Result is ERROR.
Authorization	Y	String		Authorization code from processor for original transaction.
TerminalId	N	String	1-20	Terminal Id from original transaction
MerchantId	N	String	1-20	Merchant Id from original transaction

RefData	N	Int	1-10	Additional reference data returned by host on original transaction.
CardType	Y	Fixed String	-	Type of card in original transaction. Visa/MasterCard/Amex/Diners/Discover/JCB. Based on card ranges defined in POSLynx configuration.
TranDate	Y	String	6	Date of transaction in MM/DD/YY format.
TranTime	Y	String	6	Time of transaction in HH:MM:SS format.
CardNumber	Y	Int	4	Last 4 digits of Account number from original transaction.
ExpiryDate	Y	Int	4	Expiry Date from original transaction.
Amount	Y	Float	1-7	Base Amount of original transaction.
Gratuity	N	Float	3-7	Gratuity on original transaction
SettleAmount	Y	Float	3-7	Amount settled on original transaction
TransactionType	Y	String	1-20	Type of original transaction e.g. C-Sale for Credit Sale
TransactionStatus	Y	String	1-20	Status of original transaction e.g. Approved
Receipt	N			Contains a formatted line by line receipt for simple printing.

Table 108

Example Response

```
<PLResponse>
    <Command>ITEMDETAIL</Command>
    <Result>APPROVED</Result>
    <ResultText>Transaction Approved</ResultText>
    <ErrorCode>0000</ErrorCode>
    <CardType>MasterCard</CardType>
    <CardInput>Swiped</CardInput>
    <Authorization>194372</Authorization>
    <TerminalId>001</TerminalId>
    <MerchantId>1234567890</MerchantId>
    <RecNum>1076</RecNum>
    <RefData>00000123</RefData>
    <TranDate>11/30/09</TranDate>
    <TranTime>16:06:58</TranTime>
    <CardNumber>************5454</CardNumber>
    <ExpiryDate>1212</ExpiryDate>
    <TransactionType>C-Sale</TransactionType>
    <TransactionStatus>Approved</TransactionStatus>
    <Amount>10.00</Amount>
    <SettleAmount>11.00</SettleAmount>
    <Gratuity>1.00</Gratuity>
    <Receipt>
        <Receipt1>Item Detail</Receipt1>
```

```
        <Receipt2>----------------------------------------</Receipt2>
        <Receipt3 />
        <Receipt4>Record Number: 1076</Receipt4>
        <Receipt5>Card Type: MasterCard</Receipt5>
        <Receipt6>Card #: ************5454 Exp:1212</Receipt6>
        <Receipt7>Card Input: Swiped</Receipt7>
        <Receipt8>Status: Approved</Receipt8>
        <Receipt9>Type: C-Sale</Receipt9>
        <Receipt10>Date: 11/20/09 Time: 16:06:58</Receipt10>
        <Receipt11>Amount: $10.00</Receipt11>
        <Receipt12>Gratuity Amount: $1.00</Receipt12>
        <Receipt13>Settle Amount: $11.00</Receipt13>
        <Receipt14>Auth Code: 194372</Receipt14>
        <Receipt15>Reference #: 00000123</Receipt15>
        <Receipt16>Terminal #: 001</Receipt16>
        <Receipt17>Merchant #: 1234567890</Receipt17>
    </Receipt>
</PLResponse>
```

Batch Summary by Card Type

This transaction obtains information on the current batch. Reports total sales and total count of transactions for each card type (Visa, Amex, Mastercard, Gift, etc.). Detailed data is returned in XML data fields. Also, the information is available in a report in the *<Receipt>* tags, which can be printed to a standard 42 character Receipt Printer.

Element Tag	Required	Type	Size	Notes
Command	Y	Fixed String	-	BATCHCARDTOTALS
IPAddress	N	String	7-15	Override the configured POSLynx IP Address. Allows controlling which POSLynx processes each transaction individually. Only applicable to TNP-CG.
IPPort	N	Numeric	4-5	Override the configured POSLynx port. Used with *<IPAddress>* to control which POSLynx and port are used to process each transaction. Only applicable to TNP-CG.

Table 109

Example Request

```
<PLRequest>
    <Command>BATCHCARDTOTALS</Command>
</PLRequest>
```

Element Tag	Required	Type	Size	Notes
Command	Y	Fixed String	-	BATCHCARDTOTALS
Result	Y	Fixed String	-	APPROVED/ERROR/DECLINED- result of this request. Will be declined if RecNum cannot be found.
ResultText	Y	String	1-20	Text to describe result of this request – e.g. explain error or result that could be placed on POS screen.
ErrorCode	Y	Int	1-10	Error code - 0000 if everything is successful. 1 if Result is DECLINED. Numeric error code if the Result is ERROR.
Receipt	N			Contains a formatted line by line receipt for simple printing.
Totals	Y	Int	1-10	Contains Card type totals.
Amex, Visa	N	String	1-20	Card Types. Each card type is only included if there is a transaction for that type.
Amount	Y	Float	1-7	Total amount of transactions in the current batch for the specific card type.
ItemCount		Int	1-10	Total number of transactions in the current batch for the specific card type.

Table 110

Example Response

```
<PLResponse>
    <Result>APPROVED</Result>
    <ResultText>Transaction Approved</ResultText>
    <Command>BATCHCARDTOTALS</Command>
    <Receipt>
        <Receipt1>Batch Summary by Card Type</Receipt1>
        <Receipt2>----------------------------------------</Receipt2>
        <Receipt3 />
        <Receipt4>Amex                  5          123.00</Receipt4>
        <Receipt5>Visa                  2           10.00</Receipt5>
        <Receipt6>MasterCard           20          139.40</Receipt6>
        <Receipt7>Discover              0            0.00</Receipt7>
        <Receipt8>PrivateLabel          0            0.00</Receipt8>
        <Receipt9>Diners                0            0.00</Receipt9>
        <Receipt10>JCB                   0            0.00</Receipt10>
        <Receipt11>OtherCard             0            0.00</Receipt11>
        <Receipt12>Gift                  1            2.00</Receipt12>
        <Receipt13>Loyalty               0            0.00</Receipt13>
        <Receipt14>Debit                 0            0.00</Receipt14>
        <Receipt15>EBT                   0            0.00</Receipt15>
    </Receipt>
    <Totals>
        <Amex>
            <Amount>123.00</Amount>
            <ItemCount>5</ItemCount>
        </Amex>
        <Visa>
            <Amount>10.00</Amount>
            <ItemCount>2</ItemCount>
        </Visa>
        <MasterCard>
            <Amount>139.40</Amount>
            <ItemCount>20</ItemCount>
        </MasterCard>
        <Gift>
            <Amount>2.00</Amount>
            <ItemCount>1</ItemCount>
```

```
        </Gift>
    </Totals>
</PLResponse>
```

Chapter 9: PIN Pad Functions

The content in this chapter covers only Equinox 5300/5200 series PIN Pad commands.

All functions/commands that are payment related (that is, credit, debit, gift card, or other similar transactions) have been omitted. If this information is needed, refer to the preceding sections in this document.

Overview

The XML functions/commands described in this chapter control the PIN Pad for obtaining mag stripe information and cardholder PIN information, or for displaying content/media to PIN Pad models equipped with a graphical display. Not all of these functions are needed, as typically other functions can be used more efficiently to accomplish the same task. For example, it is possible to use the CCSALE command and have it prompt for the mag stripe, rather than calling the function PPGETSTRIPE before calling the CCSALE command.

These functions are only available in certain configurations. The following must all be true:

- The PIN Pad must be either the Equinox 5300 or 5200 series.
- The PIN Pad must be connected to the POS system (not to the POSLynx).
- The POS system must be using either the TNP-CG or the TNP-CG AX application for communication with the POSLynx.
- Other configuration must be done before using these commands.

For more information, refer to the document, "Using the TransNetPOS-CG to Access the XML API for POSLynx with TransNet".

PIN Pad Functions

The XML functions/commands described in the following sections control the PIN Pad for obtaining mag stripe and PIN information. Not all of these functions are needed as other functions can be used more efficiently. For example, it is possible to use the CCSALE command and have it prompt for the mag stripe. You do not need to display additional information before the transaction, however, the screen will be left at the word "processing" and may be misleading. More information must be sent to the display to show the customer that the transaction is complete.

The full set of commands can be used only with the Equinox L5300 and L5200 series of PIN Pads. Additional configuration must be done before using these commands.

IMPORTANT–PLEASE NOTE: As long as you do not use the PPGETSWIPE command, your POS application will not be considered in scope for PCI PA-DSS. If you do use the PPGETSWIPE command, then you will be handling the card data, and consequently, will be deemed in scope.

For further details, see the document "Using the TransNetPOS-CG to Access the XML API for POSLynx for TMS".

Information pertaining to the transaction-specific elements is provided in the following sections.

NOTE: The default timeout value for the PINPad is 30 seconds. The PINPad will respond with a timeout after 30 seconds has elapsed, which overrides the configured timeout value set in the TNP-CG application if it is greater than 30 seconds.
On PINPad command timeout, POSLynx will return a result with <ErrorCode> set to 0130, <Result> set to ERROR and <ResultText> set to PINPad timeout.

Example timeout response:

```
<PLResponse>
      <ResultText>Pinpad timeout</ResultText>
      <Result>ERROR</Result>
      <ErrorCode>0130</ErrorCode>
      <Command>PPDISPLAY</Command>
</PLResponse>
```

PIN Pad Initialize

Initializes the PIN Pad and sets the welcome message.

Element Tag	Required	Type	Size	Notes
Command	Y	Fixed String	-	PPINIT
IdlePrompt	N	String	Varies by PIN Pad.	Message to display on PIN Pad when not in use.

Table 111

Example Request

```
<PLRequest>
    <Command>PPINIT</Command>
    <IdlePrompt>Welcome to our Shop</IdlePrompt>
</PLRequest>
```

Figure 2: Idle Prompt Display

Element Tag	Required	Type	Size	Notes
Command	Y	Fixed String	-	PPINIT
Result	Y	Fixed String	-	One of Success / ERROR
ResultText	Y	String	1-20	Text to describe result. For example, explain error or result that could be placed on POS screen.
ErrorCode	Y	String	1-20	Error code - 0000 if successful (1000 for L5300). Numeric error code if the Result is ERROR.

Table 112

Example Response—Success

```
<PLResponse>
    <Command>PPINIT</Command>
    <Result>Success</Result>
    <ResultText>PinPad Initialized</ResultText>
    <ErrorCode>1000</ErrorCode>
</PLResponse>
```

Example Response—Failure

```
<PLResponse>
    <Result>ERROR</Result>
    <ErrorCode>0130</ErrorCode>
    <Command>PPINIT</Command>
    <ResultText>PIN PAD FAIL</ResultText>
</PLResponse>
```

PIN Pad Initialize (EMV Only)

At times, the PIN Pad must be initialized or synchronized with the host. This command forces the initialization of the PIN Pad. If BatchIndex is omitted, the POSLynx will find the host configured for Debit and initialize to this host. In only certain cases is it necessary to specify the index.

Element Tag	Required	Type	Size	Notes
Command	Y	Fixed String	-	INITPINPAD
BatchIndex	N	Integer	1-2	Index of batch to be changed. If only one host/batch this value is 0. If more than one batch then it will depend on the POSLynx setup. There will be a command added later to determine this value programatically via the XML API.

Table 113

Example Request

```
<PLRequest>
    <Command>INITPINPAD</Command>
</PLRequest>
```

Alternatively:

```
<PLRequest>
    <Command>INITPINPAD</Command>
    <BatchIndex>1</BatchIndex>
</PLRequest>
```

Example Response

An approved response:

```
<PLResponse>
    <Result>APPROVED</Result>
    <Command>INITPINPAD</Command>
    <ResultText>PinPad Initialized</ResultText>
    <ErrorCode>0000</ErrorCode>
</PLResponse>
```

Example Response—Failure

```
<PLResponse>
      <Result>ERROR</Result>
      <ErrorCode>0130</ErrorCode>
      <Command>INITPINPAD</Command>
      <ResultText>PIN PAD FAIL</ResultText>
</PLResponse>
```

PIN Pad Reset

This command will terminate all other pending transactions for the given lane. The PIN Pad attached to the lane will be placed in the 'ready' state.

Element Tag	Required	Type	Size	Notes
Command	Y	Fixed String	-	PPRESET

Table 114

Example Request

```
<PLRequest>
    <Command>PPRESET</Command>
</PLRequest>
```

Element Tag	Required	Type	Size	Notes
Command	Y	Fixed String	-	PPRESET
Result	Y	Fixed String	-	One of Success / ERROR
ResultText	Y	String	1-20	Text to describe result. For example, explain error or result that could be placed on POS screen.
Response	Y	String	1-20	'Reset Done' on success

Table 115

Example Response—Success

```
<PLResponse>
    <Command>PPRESET</Command>
    <Result>Success</Result>
    <ResultText>Success</ResultText>
    <Response>Reset Done</Response>
</PLResponse>
```

Example Response—Failure

This command does not return any error result.

PIN Pad Reset (EMV Only)

Executing the RESETPINPAD command will activate the PINpad and acquire the CAPK public key file for all EMV card types the host supports. The chip reader on the PINpad cannot be used without this command first being executed. Should a PINpad be replaced, the RESETPINPAD command must again be executed (as the CAPK must be downloaded). In the event of a failed transaction, where the customer has only been issued a card, issuing this command will ensure the latest CAPK file has been downloaded.

Element Tag	Req	Type	Size	Notes
Command	Y	Fixed String	-	RESETPINPAD

Table 116

Example Request

```
<PLRequest>
    <Command>RESETPINPAD</Command>
</PLRequest>
```

Common Elements for Responses

Element Tag	Req	Type	Size	Notes
Command	Y	Fixed String	-	RESETPINPAD
Result	Y	Fixed String	-	One of Success or Error
ResultText	Y	String	1-20	Text to describe result – e.g. explain error or result that could be placed on POS screen.
Response	Y	String	1-20	'Reset Done' on success

Table 117

Example Response

```
<PLResponse>
     <Result>APPROVED</Result>
     <Authorization />
```

```
      <RecNum />
      <RefData>000000000000</RefData>
      <Language>English</Language>
      <ErrorCode>0000</ErrorCode>
      <AuthAmt>0.00</AuthAmt>
      <TransactionDate>130320</TransactionDate>
      <TransactionTime>175547</TransactionTime>
      <Command>RESETPINPAD</Command>
      <ResultText>INITIALIZE OK</ResultText>
      <Id>9999</Id>
      <ClientId>99</ClientId>
      <MerchantId>000318012372997</MerchantId>
     <TerminalId>01163448</TerminalId>
</PLResponse>
```

Example Response—Failure

This command does not return any error result.

Host Initialization (EMV Only - FirstData Specific)

The Data Wire ID is unique to every given TID and MID. In the event that Data Wire ID must be re-registered (for example when a "DID MISSING" pin-pad error is displayed), the HOST_INIT command should be executed (and an approved response received).

Element Tag	Req	Type	Size	Notes
Command	Y	Fixed String	-	HOST_INIT

Table 118

Example Request

```
<PLRequest>
      <Command>HOST_INIT</Command>
 </PLRequest>
```

Response Elements

Element Tag	Req	Type	Size	Notes
Command	Y	Fixed String	-	RESETPINPAD
Result	Y	Fixed String	-	One of Success / ERROR

ResultText	Y	String	1-20	Text to describe result – e.g. explain error or result that could be placed on POS screen.
Response	Y	String		'Reset Done' on success

Table 119

Example Response

```
<PLResponse>
     <Result>APPROVED</Result>
     <Authorization />
     <RecNum />
     <RefData>000000000000</RefData>
     <Language>English</Language>
     <ErrorCode>0000</ErrorCode>
     <AuthAmt>0.00</AuthAmt>
     <TransactionDate>130320</TransactionDate>
     <TransactionTime>175547</TransactionTime>
     <Command>RESETPINPAD</Command>
     <ResultText>INITIALIZE OK</ResultText>
     <Id>9999</Id>
     <ClientId>99</ClientId>
     <MerchantId>000318012372997</MerchantId>
     <TerminalId>01163448</TerminalId>
</PLResponse>
```

PIN Pad Get Signature

Displays a form on the PIN Pad allowing a customer to sign on the screen, and returns data representing the signature image.

NOTE: Function currently only supported by the Equinox L5300 Payment Terminal.

Element Tag	Required	Type	Size	Notes
Command	Y	Fixed String	-	PPGETSIGNATURE

Table 120

Example Request

```
<PLRequest>
    <Command>PPGETSIGNATURE</Command>
</PLRequest>
```

Element Tag	Required	Type	Size	Notes
Command	Y	Fixed String	-	PPGETSIGNATURE
Result	Y	Fixed String	-	One of Success / ERROR
ResultText	Y	String	1-20	Text to describe result – e.g. explain error or result that could be placed on POS screen.
Signature	Y	String	1-20	Equinox format of ASCII text to represent signature image.

Table 121

Example Response—Success

```
<PLResponse>
    <Command>PPGETSIGNATURE</Command>
    <Result>Success</Result>
    <ResultText>Success</ResultText>
    <Signature>  @$  0 +/XAOIDYJ#E"2CIATI"45D1S&?F$C"990LH64)
                *9%_""Q%<DJ26(VDK!F Z\</Signature>
</PLResponse>
```

Figure 3: Signature Screen

Example Response—Failure

Will return the following response after 30 seconds if the command fails to execute:

```
<PLResponse>
    <ResultText>Pinpad timeout</ResultText>
    <Result>ERROR</Result>
    <ErrorCode>0130</ErrorCode>
    <Command>PPGETSIGNATURE</Command>
</PLResponse>
```

> **NOTE:** The same response is returned should the user fail to make a selection after 30 seconds.

PIN Pad Get Cash Back

This function displays a form allowing a customer to enter an amount of cash-back desired. The selected amount is returned in the response. The form has a preset amount with values of $10, $20, $40, $60, $80, $100, $150, $200.

> **NOTE:** Function currently only supported by the Equinox L5300 Payment Terminal.

Element Tag	Required	Type	Size	Notes
Command	Y	Fixed String	-	PPGETCASHBACK

Table 122

Example Request

```
<PLRequest>
    <Command>PPGETCASHBACK</Command>
</PLRequest>
```

Element Tag	Required	Type	Size	Notes
Command	Y	Fixed String	-	PPGETCASHBACK
Result	Y	Fixed String	-	One of Success / ERROR
ResultText	Y	String	1-20	Text to describe result. For example, explain error or result that could be placed on POS screen.
ErrorCode	N	Int	4	Numeric error code if the Result is ERROR.
Response	Y	String	1-20	The amount requested as Cashback, e.g. 10.00, 20.00 etc.

Table 123

Example Response—Success

```
<PLResponse>
    <Command>PPGETCASHBACK</Command>
    <Result>Success</Result>
    <ResultText>Success</ResultText>
    <Response>$20.00</Response>
</PLResponse>
```

Figure 4: Get Cash Back display

Example Response—Failure

Will return the following response if the command fails to execute after 30 seconds:

```
<PLResponse>
    <ResultText>Pinpad timeout</ResultText>
    <Result>ERROR</Result>
    <ErrorCode>0130</ErrorCode>
    <Command>PPGETCASHBACK</Command>
</PLResponse>
```

> **NOTE:** The same response is returned should the user fail to make a selection after 30 seconds.

PIN Pad Show Item List

This function displays a list of all the individual items being purchased with total purchase and tax amounts. If no items have been added to the list, it will return an empty item list form with total tax and total purchase amounts set to $0.00. PPADDITEM command will add items to the list and update the total.

> **NOTE:** Currently only supported by the Equinox L5300 Payment Terminal.

Element Tag	Required	Type	Size	Notes
Command	Y	Fixed String	-	PPITEMLIST

Table 124

Example Request

```
<PLRequest>
    <Command>PPITEMLIST</Command>
</PLRequest>
```

Element Tag	Required	Type	Size	Notes
Command	Y	Fixed String	-	PPITEMLIST
Result	Y	Fixed String	-	One of Success / ERROR
ResultText	Y	String	1-20	Text to describe result. For example, explain error or result that could be placed on POS screen.
ErrorCode	Y	String	4	Error code - 0000 if everything is successful. Numeric error code if the Result is ERROR.

Table 125

Example Response—Success

```
<PLResponse>
    <Command>PPITEMLIST</Command>
    <Result>Success</Result>
    <ResultText>Success</ResultText>
    <ErrorCode>0000</ErrorCode>
</PLResponse>
```

Example Response—Failure

```
<PLResponse>
```

```
     <ResultText>Pinpad timeout</ResultText>
    <Result>ERROR</Result>
    <ErrorCode>0130</ErrorCode>
    <Command>PPITEMLIST</Command>
</PLResponse>
```

> **NOTE:** To ensure minimal wait time, a successful response is only an indication that the POSLynx received the request from the ECR, but **does not indicate** the PIN Pad received and successfully processed the request.

PIN Pad Add Item to List

This function adds an item to a list displayed in a form, or edits an item in a list. PPADDITEM must be used after the PPITEMLIST command. The tax amount and total purchase amount including tax, is updated with running totals of items displayed on the screen.

> **NOTE:** Currently only supported by the Equinox L5300 Payment Terminal.

Itemid

Each item is added to the screen, with a scroll bar included when the list exceeds the length of the screen. Items can be added in any order, but are displayed based on the sorted order of the <*Itemid*> number. Consider the example scenario where *ItemId 002* was added before *ItemId 001*; it is *ItemId 001* which will be displayed at the top of the screen.

Sub Items

Sub items are descriptors to the higher level item, and they are indented on the screen in two (2) levels. Level 2 are in the form 001.001 and level 3 are in the form 001.001.001. These levels are also sorted, so the ItemId 001.002 will always appear after ItemId 001.001 on screen, regardless of order added. If sub items are not used, 000 can be used, or the field ignored.

For example:

Specifying an <*ItemId*> of 001.000.000 is the same as 001 - both will be top level items.

Specifying an <*ItemId*> of 001.001.000 is the same as 001.001 - both will be second level items.

Up to a maximum of 8 digits can be used for each section (ie. 1.1.1 or 01.01.01 or 001.001.001 or 0001.0001.0001). However, they are sorted as numbers and will display in that sorted order. Therefore 001 and 01 are the same number and the sort order will be

unpredictable. It is acceptable to mix Level indicator lengths as well (for example, 001.01.02).

A top level item will always show the quantity and the price/amount. Second and third level items will not show quantity if zero or blank, and no price will be shown if it is 0, 0.00, or blank.

Tax

The <*Tax*> element is always the new Total tax for all items on the screen, and is added into the Total. If an item is added or removed from the list, the amount sent in the <*Tax*> tag will be the new Total tax. This will be added to the current total of items when the total is listed on the bottom of the screen.

To summarize, the submitted value for <*Tax*> is what is displayed on the item list form as the tax amount. The total purchase amount displayed on the form is the sum of all item amounts and the <*Tax*> value.

Edit an Item

To edit an item that is already present, PPADDITEM is sent, with the same ItemId as a previous item and it will be updated on screen. If a field, such as Amount, is changed, the old amount is saved and will be removed and the new price is added to the total.

Element Tag	Required	Type	Size	Notes
Command	Y	Fixed String	-	PPADDITEM
Item	Y	String	1-20	Text description of Item
Tax	N	Float	3-7	Total tax on this order for all items. Not just tax on this item
Amount	Y	Float	1-7	Extended Cost of line Item, if quantity is greater than 1, this will be the total cost.
ItemId	Y	String	1-20	Order of item on display screen, see above
PluNumber	N	String	1-20	Item merchant PLU number.
SerialNo	N	String		Serial Number of the item. Optional, as tracking of specific items sold may not be desired, but allows tracking of a single specific item if required.
Quantity	N	Int	1-5	The number of items purchased on this line item
Department	N	String	1-20	Not displayed, sent to NetVu for sorting transaction details
Id	N	String	1-20	Id to associate this item to – typically an invoice number. Will be used to associate this item with a transaction (e.g. credit card or debit card transaction)

Table 126

Example Requests

Item level 1 request:

```
<PLRequest>
```

```
    <Command>PPADDITEM</Command>
    <Item>Kohler Chrome Accents</Item>
    <Tax>24.45</Tax>
    <Amount>99.00</Amount>
    <ItemId>001</ItemId>
</PLRequest>
```

Figure 5: PIN Pad Add Item to List (level 1)

Item level 2 request:

```
<PLRequest>
    <Command>PPADDITEM</Command>
    <Item>Copper Tips</Item>
    <Tax>24.45</Tax>
    <Amount>99.00</Amount>
    <ItemId>001.001</ItemId>
</PLRequest>
```

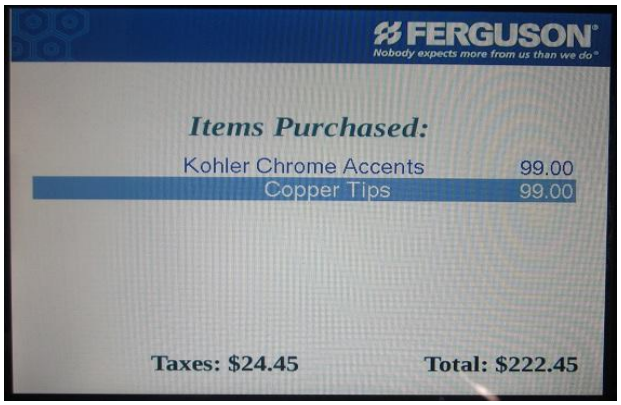


Figure 6: PIN Pad Add Item to List (level 2)

Item level 3 request:

```
<PLRequest>
    <Command>PPADDITEM</Command>
    <Item>Included Free</Item>
    <Tax>24.45</Tax>
    <Amount>99.00</Amount>
    <ItemId>001.001.001</ItemId>
</PLRequest>
```

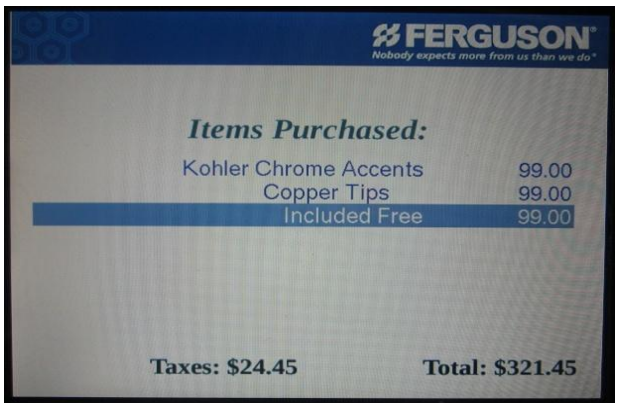


Figure 7: PIN Pad Add Item to List (level 3)

Element Tag	Required	Type	Size	Notes
Command	Y	Fixed String	-	PPADDITEM
Result	Y	Fixed String	-	One of Success / ERROR
ResultText	Y	String	1-20	Text to describe result – e.g. explain error or result that could be placed on POS screen.
ErrorCode	Y	String	4	Numeric error code. 0000 if no error occurred.
ItemId	Y	String	1-20	Same value which was submitted in the request

Table 127

Example Response—Success

```
<PLResponse>
    <Command>PPADDITEM</Command>
    <Result>Success</Result>
    <ResultText>Success</ResultText>
    <ErrorCode>0000</ErrorCode>
    <ItemId>001</ItemId>
```

```
</PLResponse>
```

The new total is automatically computed as the sum of all <*Amounts*> from all PPADDITEM commands and the last submitted <*Tax*> value.

Example Response—Failure

This command may return an error when it is submitted to POSLynx using the TNP-CG application.

> **NOTE:** To ensure minimal wait time, a successful response is only an indication that the POSLynx received the request from the ECR. It **does not indicate** the PIN Pad received and successfully processed the request.

PIN Pad Delete Item on List

This command deletes an existing item on the form. This command must be used after PPITEMLIST and PPADDITEM commands.

> **NOTE:** Function currently only supported by the Equinox L5300 Payment Terminal.

The item number must match a previous item number that was sent with the PPADDITEM command. For example, if item 002.003.001 was previously added, it can be deleted by specifying that particular <*ItemId*> in the PPDELITEM command.

There is no relationship between a top level item and second and third level items. Therefore, should 001.000.000, 001.001.000 and 001.001.001 be added (first, second, and third level items) and then 001.000.000 is subsequently deleted, the two 'children' items will remain. Three PPDELITEM commands must be issued to delete all three of these items.

Element Tag	Required	Type	Size	Notes
Command	Y	Fixed String	-	PPDELITEM
Item	Y	String	1-20	Text description of Item
Amount	Y	Float	1-7	Amount that the total is reduced by. This may be different that the original amount.
ItemId	Y	String	1-20	Order of item on display screen, see above
Tax	N	Float	3-7	Total tax on this order for all items

Table 128

Example Request

```
PLRequest>
     <Command>PPDELITEM</Command>
     <Item>Kohler Crome Accents</Item>
```

```
    <Tax>16.53</Tax>
    <Amount>99.00</Amount>
    <ItemId>001</ItemId>
</PLRequest>
```

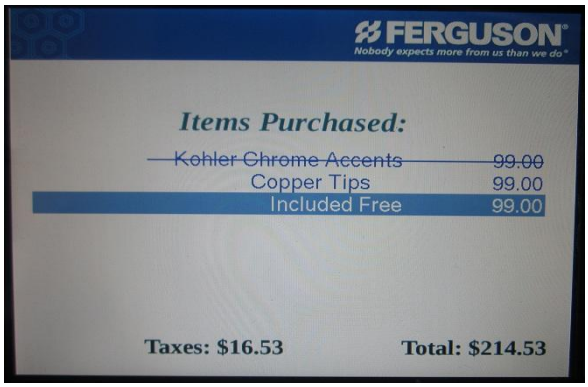


Figure 8: PIN Pad Delete Item on List

Example Response—Success

```
<PLResponse>
    <Command>PPDELITEM</Command>
    <Result>Success</Result>
    <ResultText>Success</ResultText>
    <Response>Item Deleted</Response>
    <ItemId>001</ItemId>
</PLResponse>
```

If the item to delete does not exist, the result will be ERROR – PPDELITEM FAILURE (0130).

> **NOTE:** Removing the line item does not automatically remove all the items in levels below it. They must be removed manually. Totals will be updated accordingly along with the new tax value taken from the last request.

Example Response—Failure

Will return the following response after 30 seconds if the command fails to execute:

```
<PLResponse>
    <ResultText>Pinpad timeout</ResultText>
    <Result>ERROR</Result>
    <ErrorCode>0130</ErrorCode>
    <Command>PPDELITEM</Command>
```

```
</PLResponse>
```

Submitting improper values in the request will yield this result:

```
<PLResponse>
    <ResultText>PPDELITEM FAILURE</ResultText>
    <Result>ERROR</Result>
    <ErrorCode>0130</ErrorCode>
    <Command>PPDELITEM</Command>
</PLResponse>
```

PIN Pad Remove Item on List

This command removes an existing item on the form. Similar to PPDELITEM except an item is removed rather than stroked-out. This command must be used after PPITEMLIST and PPADDITEM commands.

> **NOTE:** Function currently only supported by the Equinox L5300 Payment Terminal.

The item number must match a previous item number that was sent with the PPADDITEM or PPDELITEM command. For example, if item 002.003.001 was previously added, it can be removed by specifying that particular <ItemId> in the PPRMVITEM command.

There is no relationship between a top level item and second and third level items. Therefore, should 001.000.000, 001.001.000 and 001.001.001 be added (first, second, and third level items) and then 001.000.000 is subsequently removed, the two 'children' items will remain. Three PPRMVITEM commands must be issued to remove all three of these items.

Element Tag	Required	Type	Size	Notes
Command	Y	Fixed String	-	PPRMVITEM
Amount	Y	Float	-	Amount that the total is reduced by. This may be different that the original amount.
ItemId	Y	String		Order of item on display screen, see above.
Tax	N	Float		Total tax on this order for all items.

Table 129

Example Request

```
<PLRequest>
      <Command>PPRMVITEM</Command>
      <Item>Kohler Crome Accents</Item>
      <Tax>16.53</Tax>
      <Amount>99.00</Amount>
      <ItemId>001</ItemId>
      </PLRequest>
```

Example Response—Success

```
<PLResponse>
      <Command>PPRMVITEM</Command>
      <Result>Success</Result>
      <ResultText>Success</ResultText>
      <Response>Item Removed</Response>
      <ItemId>001</ItemId>
</PLResponse>
```

If the item to delete does not exist, the result will be ERROR – ItemId does not exist. ErrorCode 2024.

> **NOTE:** Removing the line item does not automatically remove all the items in levels below it. They must be removed manually. Totals will be updated accordingly along with the new tax value taken from the last request.

Example Response—Failure

Will return the following response after 3 seconds if the command fails to execute:

```
<PLResponse>
      <ResultText>Pinpad timeout</ResultText>
      <Result>ERROR</Result>
      <ErrorCode>0130</ErrorCode>
      <Command>PPRMVITEM</Command>
      <ItemId>001</ItemId>
      <Response>ERROR</Response>
</PLResponse>
```

PIN Pad Display a Message

This command uses a form to show a custom message and/or collect a customer's response.

The <*Text*> element contains the text string to be displayed.

The '|' character separates lines and can be used to create blank lines.

The <*Lines*> element is matched to a form that will accept values of '1', '2', '4', or '7'. The font on the screen will scale according to the number of lines presented. A small number of lines (for example, 1) will use a larger font to fill up the screen, while a display with 7 lines will use a smaller font in order to accommodate more text on the screen. Text is centered and will not be 'auto-sized' on-screen. Should the <*Lines*> element not be provided, the value will default to 7.

The optional <*Buttons*> element, when provided, will allow buttons to be displayed on the screen. Up to three string values can be provided for the <*Buttons*> element, each separated by a "|" resulting in 1, 2, or 3 buttons being displayed at the bottom of the screen (with labels as defined by the <Button> tag). The response will contain the name of the button pressed.

To display two (2) buttons, one on the left and one on the right side of the screen, a double separator is inserted.

For example:

```
<Buttons>OK||Cancel</Buttons>
```

A response will be returned based on the button pressed, the response being the text of the pressed button in question. It should be noted that this form will remain on the screen after the button is clicked.

Therefore, to avoid confusion and unexpected results, a second PPDISPLAY should be sent to remove the buttons from the page. A maximum of 200 characters can be sent.

The standard font for this screen is Arial, but this can be modified to a different font style. This can be generated as a new form download.

Element Tag	Required	Type	Size	Notes
Command	Y	Fixed String	-	PPDISPLAY
Lines	N	Integer	1	Number of lines on display, can be 1, 2, 4, or 7
Text	Y	String	1-20	Text centered on screen, with "|" character used as line feed.
Buttons	N	String	1-20	As described in above paragraph.

Table 130

Example Request—one line display

```
<PLRequest>
    <Command>PPDISPLAY</Command>
    <Lines>1</Lines>
    <Text>Please Swipe your Card...</Text>
</PLRequest>
```

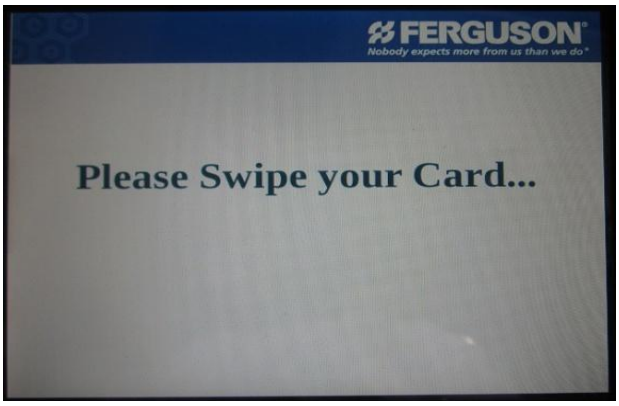


Figure 9: PIN Pad Display a Message, one line display

Example Request—two lines, one button

```
<PLRequest>
    <Command>PPDISPLAY</Command>
    <Lines>2</Lines>
    <Text>You Can print|26 char wide</Text>
    <Buttons>Button One</Buttons>
</PLRequest>
```

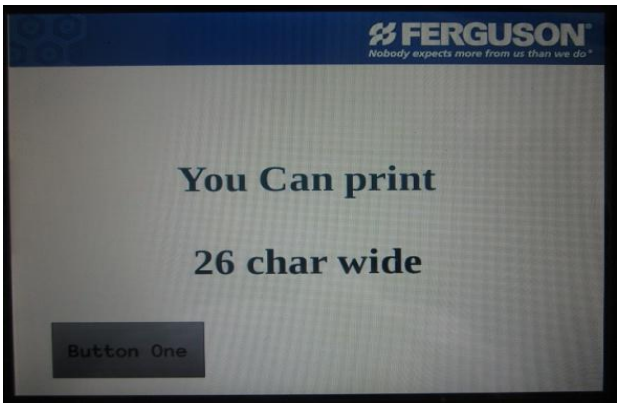


Figure 10: PIN Pad Display a Message, two lines and one button

Example Request—four lines, two buttons

```
<PLRequest>
    <Command>PPDISPLAY</Command>
    <Lines>4</Lines>
    <Text>You can print as wide as|32 characters
       across the screen|This is a the third line
       of text here.|This is the fourth line of
       text here.</Text>
    <Buttons>Button One||Button Two</Buttons>
</PLRequest>
```

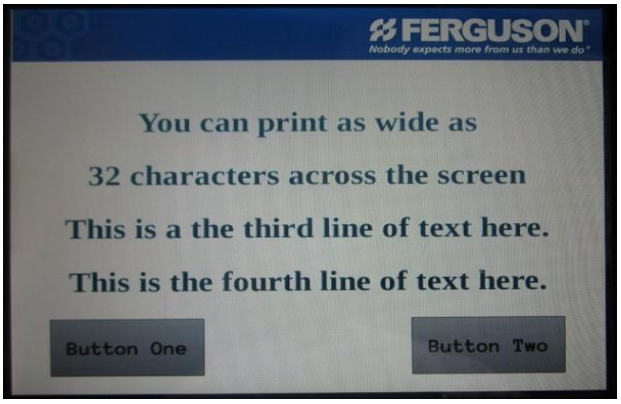


Figure 11: PIN Pad Display a Message, four lines and two buttons

Example Request—seven lines, 3 buttons

```
<PLRequest>
    <Command>PPDISPLAY</Command>
    <Lines>7</Lines>
    <Text>You can print up to 45 chararacters|Across
      the screen. You will find you are|only going
      to be limited|by the maximum number of
      characters|we give you which is 250 characters|
      This should be a lot of text</Text>
    <Buttons>Accept|Decline|Maybe</Buttons>
</PLRequest>
```

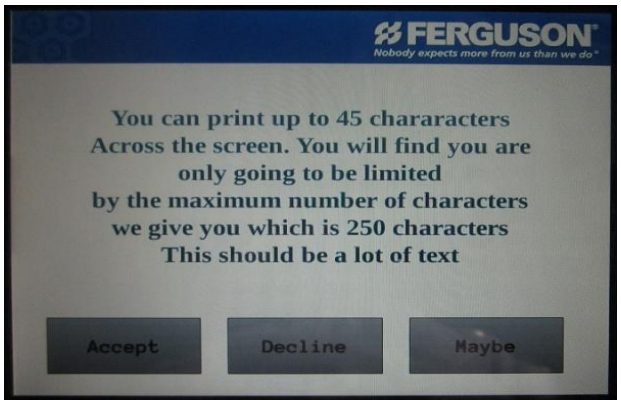


Figure 12: PIN Pad Display a Message, seven lines and three buttons

Element Tag	Required	Type	Size	Notes
Command	Y	Fixed String	-	PPDISPLAY
Result	Y	Fixed String	-	One of Success / ERROR
ResultText	Y	String	1-20	Text to describe result – e.g. explain error or result that could be placed on POS screen.
Response	Y	String	1-20	Contains the text of the button pressed. Contains 'No Buttons' if 'Buttons' not present.

Table 131

Example Response—no buttons pressed

```
<PLResponse>
    <Command>PPDISPLAY</Command>
    <Result>Success</Result>
    <ResultText>Success</ResultText>
    <Response>No Buttons</Response>
</PLResponse>
```

Example Response—OK button pressed

```
<PLResponse>
    <Command>PPDISPLAY</Command>
```

```
    <Result>Success</Result>
    <ResultText>Success</ResultText>
    <Response>OK</Response>
</PLResponse>
```

Example Response—Failure

Will return the following response after 30 seconds if the command fails to execute:

```
<PLResponse>
    <ResultText>Pinpad timeout</ResultText>
    <Result>ERROR</Result>
    <ErrorCode>0130</ErrorCode>
    <Command>PPDISPLAY</Command>
</PLResponse>
```

> **NOTE:** The <*Text*> and <*Buttons*> tags are limited to 200 characters of text. <*Buttons*> text has precedence over text in the <*Text*> tag. Text in the <*Text*> tag is truncated until the combined length of text in <*Text*> and <*Buttons*> equals 200 characters.

PIN Pad Custom Screen Display

This command displays a form that is present on the device (that is, it has been pre-loaded from an existing forms package).

> **NOTE:** Currently only supported by the Equinox L5300 Payment Terminal.

The <*FORMID*> *element* denotes the ID of a custom form already loaded on the L5300 device to be displayed on the screen.

Element Tag	Required	Type	Size	Notes
Command	Y	Fixed String	-	PPCUSTOMDISPLAY
FORMID	Y	Int	1-10	The ID of the custom form to be displayed
Text	N	String	1-20	Text centered on screen, with "|" character used as line feed
Buttons	N	String	1-20	As with PPDisplay, limited to 10 buttons.

Table 132

The <*Buttons*> element is applicable only to those forms which support button input. If the tag for <*Buttons*> is filled, the command will wait until there is a button press, or the timeout expires.

Example Request—button press

```
<PLRequest>
    <Command>PPCUSTOMDISPLAY</Command>
    <FORMID>100</FORMID>
    <Text>Show upto 210 chars(incl Button text)|7 lines|10
       Buttons|CAREFULL!|delete       buttons that might|flow over the
       text|Because you can't see them</Text>
    <Buttons>But1||But3|But4||But6|But7|But8|But9</Buttons>
</PLRequest>
```

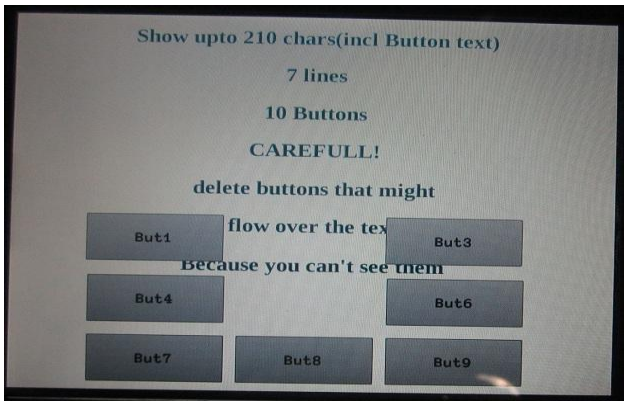


Figure 13: PIN Pad Custom Display

Example Response

```
<PLResponse>
    <Command>PPCUSTOMDISPLAY</Command>
    <Result>Success</Result>
    <ResultText>Success</ResultText>
    <Response>But3</NoButtons>
</PLResponse>
```

Example Request

```
<PLRequest>
    <Command>PPCUSTOMDISPLAY</Command>
    <FORMID>4</FORMID>
</PLRequest>
```

Figure 14: PIN Pad Custom Display, Form ID

Element Tag	Required	Type	Size	Notes
Command	Y	Fixed String	-	PPCUSTOMDISPLAY
Result	Y	Fixed String	-	One of Success / ERROR
ResultText	Y	String	1-20	Text to describe result – e.g. explain error.
Response	Y	String	1-20	The value of <*Response*> is 'No Buttons' for the case where the submitted PLRequest does not contain a <*Buttons*> element. Otherwise, the value of the <*Buttons*> pressed.

Table 133

Example Response—Success

```
<PLResponse>
    <Command>PPCUSTOMDISPLAY</Command>
    <Result>Success</Result>
    <ResultText>Success</ResultText>
    <Response>No Buttons</NoButtons>
</PLResponse>
```

Example Response—Failure

Submitting a FORMID value for a custom form which does not exist will result in this response:

```
<PLResponse>
    <Command>PPCUSTOMDISPLAY</Command>
    <Result>Success</Result>
    <ResultText>Success</ResultText>
    <Response>No Buttons</Response>
</PLResponse>
```

Will return the following response after 30 seconds if the command fails to execute:

```
<PLResponse>
    <ResultText>Pinpad timeout</ResultText>
    <Result>ERROR</Result>
    <ErrorCode>0130</ErrorCode>
    <Command>PPCUSTOMDISPLAY</Command>
</PLResponse>
```

> **NOTE:** The <*Text*> and <*Buttons*> tags are limited to 200 characters of text. <*Buttons*> text has precedence over text in the <*Text*> tag. Text in the <*Text*> tag is truncated until the combined length of text in <*Text*> and <*Buttons*> equals 200 characters.

> **NOTE:** To ensure minimal wait time, a successful response is only an indication that the POSLynx received the request from the ECR. It **does not indicate** that the PIN Pad received and successfully processed the request.

PIN Pad Upload a Form

This command uploads a forms package to the device from the Precidia Form Server. A forms package can contain numerous forms, which are comprised of both static and dynamic GUI elements (for example, labels; pictures; input boxes; videos; multi-image clips).

> **NOTE:** Currently only supported by the Equinox L5300 Payment Terminal.

<ServerIP> denotes the IP address of the Precidia Forms Server (the host for all Precidia-developed Equinox L5300 form packages).

<FormName> denotes the name of the form package required to be downloaded to the Equinox L5300.

Special Forms

There are Special Forms that perform various functions. These forms can be downloaded via NetVu, but we also provide the ability to download directly to the PIN Pad using these commands.

KEYFORM: This form will activate a session to the Equinox RKI server to download a PIN debit key. Equinox must be informed of the Serial Number of the PIN Pad and the version number. When this command is sent, the PIN Pad will restart and ask if you want to delete the script after downloading. Reply should be yes, or else it will download again.

APPLICATION: This form (ask Precidia for the form name) is used to update the O/S and FPE versions on the PIN Pad. This can be a very large file, and it is important that during the download, there are no connection failures, or else the terminal can be lost.

> **NOTE:** A form may take a long time to download, so it is generally recommended that it be done in off peak business hours. If a form download fails, you can always restart it. It will restart where it left off.

Element Tag	Required	Type	Size	Notes
Command	Y	Fixed String	-	PPUPLOADFORM
ServerIP	Y	String	1-20	Precidia Forms IP Address
FormName	Y	String	1-20	Form package to be uploaded

Table 134

Example Request

```
<PLRequest>
    <Command>PPUPLOADFORM</Command>
    <ServerIP>173.195.60.141</ServerIP>
    <FormName>feitf3b</FormName>
</PLRequest>
```

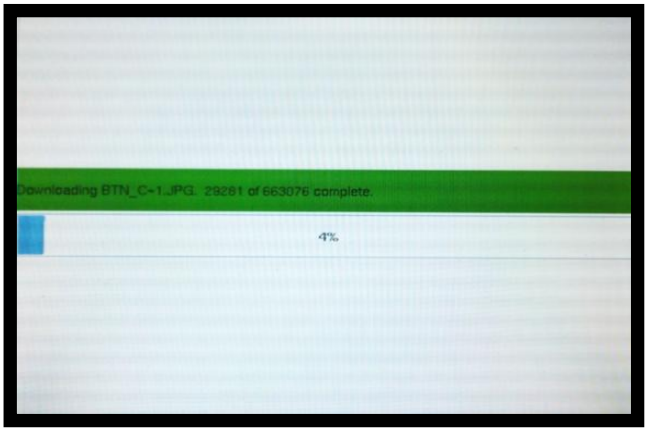


Figure 15: PIN Pad Upload a Form

Element Tag	Required	Type	Size	Notes
Command	Y	Fixed String	-	PPUPLOADFORM
Result	Y	Fixed String	-	One of Success / ERROR
ResultText	Y	String	1-20	Text to describe result – e.g. explain error.
ErrorCode	N	String	4	0000 if successful, numeric error code if the Result is ERROR.

Table 134

Example Response

```
<PLResponse>
    <Command>PPUPLOADFORM</Command>
    <Result>Success</Result>
    <ResultText>Success</ResultText>
</PLResponse>
```

Example Response—Failure

Submitting improper values in the request will yield this result:

```
<PLResponse>
    <ResultText>PinPadTimeOut</ResultText>
    <Result>ERROR</Result>
    <ErrorCode>0130</ErrorCode>
    <Command>PPUPLOADFORM</Command>
</PLResponse>
```

PIN Pad Get Swipe

This command will allow a user to perform a card swipe which will extract Track1, Track2, and Track3 information and return it to the merchant.

> **NOTE:** Currently only supported by the Equinox L5300 Payment Terminal.

Element Tag	Required	Type	Size	Notes
Command	Y	Fixed String	-	PPGETSWIPE
TextBefore	N	String	1-20	If supplied, will be used as a prompt on the PIN Pad when card swipe is being requested.
TextAfter	N	String	1-20	If supplied, will be used as a prompt on the PIN Pad after card swipe has occurred. If not supplied, defaults to the value 'Processing…'

Table 135

Example Request

```
<PLRequest>
    <Command>PPGETSWIPE</Command>
    <TextBefore>Swipe your Card</TextBefore>
    <TextAfter>Processing ....</TextAfter>
</PLRequest>
```

When the command is submitted, the first screen will display and the LEDs will indicate where to swipe your card. After, when the Swipe is successful the second screen will display.

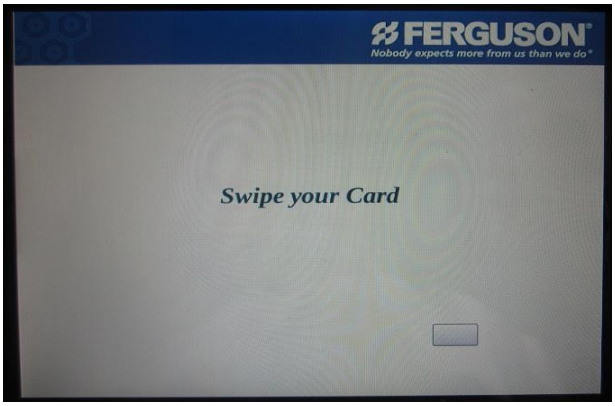


Figure 16: PIN Pad Card Swipe waiting

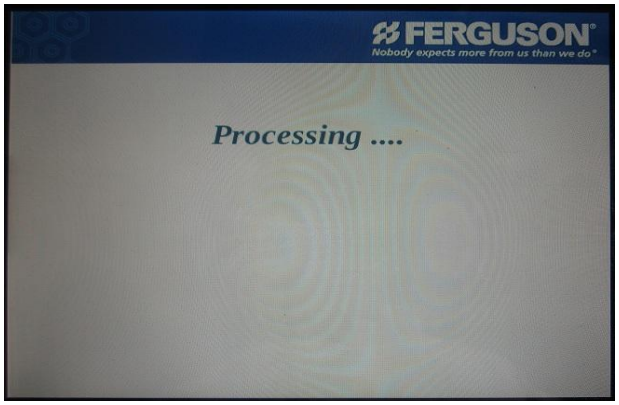


Figure 17: PIN Pad Card Swipe processing

Element Tag	Required	Type	Size	Notes
Command	Y	Fixed String	-	PPGETSWIPE
Result	Y	Fixed String	-	One of Success / ERROR
ResultText	Y	String	1-20	Text to describe result – e.g. explain error.
Track1Data	N	String	1-40	The Track1 data returned from the card swipe if available
Track2Data	N	String	1-80	The Track2 data returned from the card swipe if available
Track3Data	N	String	1-20	The Track3 data returned from the card swipe if available

Table 136

Example Response—Success

```
<PLResponse>
    <Command>PPGETSWIPE</Command>
    <Result>Success</Result>
    <ResultText>Success</ResultText>
    <Response>Success</Response>
    <Track1Data>B4788250000028291^PAYMENTECH^1012101543
         2112345678</Track1Data>
    <Track2Data>4788250000028291=10121015432112345678</Track2Data>
</PLResponse>
```

Example Response—Failure

Will return the following response after 30 seconds if the command fails to execute, or the user fails to swipe their card:

```
<PLResponse>
    <ResultText>Pinpad timeout</ResultText>
    <Result>ERROR</Result>
    <ErrorCode>0130</ErrorCode>
    <Command>PPGETSWIPE</Command>
</PLResponse>
```

Track1Data, Track2Data, Track3Data tags only contain values when the swiped card contained data for the given track.

PIN Pad Display Survey Form

This command will present the user with a survey form, allowing the user to provide their input from a list of choices (button selections).

> **NOTE:** Currently only supported by the Equinox L5300 Payment Terminal.

Gathering survey information is performed in two (2) steps. The first step is the presentation of the survey form with PPSURVEY. The second step is retrieving the result with PPLOYALTYQUERY. The PPSURVEY is considered 'active' as long as the timeout value, *<SurveyTimeOut>*, has not been exceeded. A PPLOYALTYQUERY command can be submitted to the terminal provided the *<SurveyTimeOut>* value has not been exceeded.

Element Tag	Required	Type	Size	Notes
Command	Y	Fixed String	-	PPSURVEY
SurveyID	Y	Int	1-10	The ID of the survey form which is to be presented to the user
SurveyTimeOut	Y	Int	1-10	Defaults to a value of 60 (seconds) if omitted.

Table 136

Example Request

```
<PLRequest>
    <Command>PPSURVEY</Command>
    <SurveyID>1</SurveyID>
    <SurveyTimeOut>60</SurveyTimeOut>
</PLRequest>
```

Element Tag	Required	Type	Size	Notes
Command	Y	Fixed String	-	PPSURVEY
Result	Y	Fixed String	-	Contains 'Success'
ResultText	Y	Fixed String	-	Text to describe result – 'Success'

Table 137

Example Response—Success

```
<PLResponse>
    <Command>PPSURVEY</Command>
    <Result>Success</Result>
    <ResultText>Success</ResultText>
</PLResponse>
```

Example Response—Failure

It does not return any response if the command fails to execute (that is, infinite wait). The customer should manage the timeout.

No error is returned if the value of <*SurveyId*> is invalid.

> **NOTE:** To ensure minimal wait time, a successful response is only an indication that the POSLynx received the request from the ECR. It **does not indicate** the PIN Pad received and successfully processed the request.

PIN Pad Display Reward Form

This command will present the user with a Reward Form allowing the user to provide their input from a list of choices (button selections).

> **NOTE:** Currently only supported by the Equinox L5300 Payment Terminal.

Element Tag	Required	Type	Size	Notes
Command	Y	Fixed String	-	PPREWARD
RewardID	Y	Int	1-10	The ID of the reward form which is to be presented to the user
RewardTimeOut	Y	Int	1-10	Default to value of 60 if omitted

Table 138

Example Request

```
<PLRequest>
    <Command>PPREWARD</Command>
    <RewardID>1</RewardID>
    <RewardTimeOut>60</RewardTimeOut>
</PLRequest>
```

Figure 18: PIN Pad Reward Form

Element Tag	Required	Type	Size	Notes
Command	Y	Fixed String	-	PPREWARD
Result	Y	Fixed String	-	Contains 'Success'
ResultText	Y	Fixed String	-	Text to describe result – 'Success'

Table 139

Example Response—Success

```
<PLResponse>
    <Command>PPREWARD</Command>
    <Result>Success</Result>
    <ResultText>Success</ResultText>
</PLResponse>
```

Example Response—Failure

Does not return any response if the command fails to execute (ie. Infinite wait). Customer should manage the timeout.

No error is returned if the value of <*RewardId*> is invalid.

> **NOTE:** A successful response is only an indication that the POSLynx received the request from the ECR. It **does not indicate** the PIN Pad received and successfully processed the request.

To get the response to what was entered, you will need to send a PPQUERY message next.

PIN Pad Display Claim Reward Form

This command presents a screen in which the user can enter a code or coupon number to claim a reward. The command uses the <*ClaimRewardID*> tag from the forms package.

> **NOTE:** Currently only supported by the Equinox L5300 Payment Terminal.

Element Tag	Required	Type	Size	Notes
Command	Y	Fixed String	-	PPCLAIMREWARD
ClaimRewardID	Y	Int	1-10	This is the reward form ID downloaded in the package

Table 140

Example Request

```
<PLRequest>
    <Command>PPCLAIMREWARD</Command>
    <ClaimRewardID>1</ClaimRewardID>
</PLRequest>
```

Figure 19: PIN Pad Claim Reward Display

The screen will wait until the number is entered and then either the DONE button is pressed, or the green OK button is pressed.

Element Tag	Required	Type	Size	Notes
Command	Y	Fixed String	-	PPCLAIMREWARD
Result	Y	Fixed String	-	Contains 'Success'
ResultText	Y	Fixed String	-	Text to describe result – 'Success'

Table 141

Example Response—Success

```
<PLResponse>
    <Command>PPCLAINREWARD</Command>
    <Result>Success</Result>
    <ResultText>5551212</ResultText>
</PLResponse>
```

Example Response—Failure

Will return the following response after 30 seconds if the command fails to execute, the user fails to complete the data entry, or the *<ClaimRewardID>* refers to a form which does not exist:

```
<PLResponse>
    <ResultText>Pinpad timeout</ResultText>
    <Result>ERROR</Result>
    <ErrorCode>0130</ErrorCode>
    <Command>PPCLAIMREWARD</Command>
</PLResponse>
```

PIN Pad Get Survey Result Form

This command will retrieve the results of the Display Survey Form

> **NOTE:** Currently only supported by the Equinox L5300 Payment Terminal.

Element Tag	Required	Type	Size	Notes
Command	Y	Fixed String	-	PPLOYALTYQUERY

Table 142

Example Request

```
<PLRequest>
    <Command>PPLOYALTYQUERY</Command>
    <Result>Success</Result>
    <ResultText>Success</ResultText>
    <CurrentScreeN>Survey</CurrentScreen>
    <Status>Active</Status>
</PLRequest>
```

Element Tag	Required	Type	Size	Notes
Command	Y	Fixed String	-	PPLOYALTYQUERY
Result	Y	String	1-20	Contains 'Success'
ResultText	Y	String	1-20	Text to describe result – 'Success'
CurrentScreen	Y	String	1-20	Survey – If the originating command was PPSURVEY
Status	Y	Fixed String	-	Active – if the timeout period has not expired; Timedout – if the active period has expired with no feedback from the user; Otherwise – will contain the text of the button which was selected

Table 143

Example Response—Success

```
<PLRequest>
    <Command>PPLOYALTYQUERY</Command>
```

```
    <Result>Success</Result>
    <ResultText>Success</ResultText>
    <CurrentScreeN>Survey</CurrentScreen>
    <Status>Active</Status>
</PLRequest>
```

Example Response—Failure

This command does not return any error result.

PIN Pad Reset Statistics

The terminal monitors over 100 different items, which may be of interest to the merchant (for example, the number of successful card swipes, number of failed card swipes, etc.) This command clears all internal statistical counters within the payment terminal.

If the Generate PIN Pad Statistics feature is enabled, statistical information is sent from the payment terminal to NetVu every hour. Normally, NetVu will clear the counters once the statistical information has been received, and hence, this command is usually not required.

> **NOTE:** Currently only supported by the Equinox L5300 Payment Terminal.

Element Tag	Required	Type	Size	Notes
Command	Y	Fixed String	-	PPRESETSTATS

Table 144

Example Request

```
<PLRequest>
    <Command>PPRESETSTATS</Command>
</PLRequest>
```

Element Tag	Required	Type	Size	Notes
Command	Y	Fixed String	-	PPRESETSTATS
Result	Y	Fixed String	-	One of Success / ERROR
ResultText	Y	String	1-20	Pinpad Statistics Reset Success/Failure

Table 145

Example Response—Success

```
<PLResponse>
    <Command>PPRESETSTATS</Command>
    <Result>Success</Result>
    <ResultText>Pinpad Statistics Reset Success</Command>
</PLResponse>
```

Example Response—Failure

```
<PLResponse>
    <ResultText>Pinpad Statistics Reset Failure</ResultText>
    <Result>ERROR</Result>
    <ErrorCode>0130</ErrorCode>
    <Command>PPRESETSTATS</Command>
</PLResponse>
```

Chapter 10: Acronyms

AEIPS	American Express ICC Payment Specification
AVS	Address Verification System
API	Application Programming Interface
CSC	Card Security Code
CVC	Card Verification Code
CVD	Card Verification Data
CVV	Card Verification Value
EBT	Electronic Benefit Transfer
ECR	Electronic Cash Register
EMV	Europay, MasterCard & Visa
MSR	Magnetic Stripe Reader
PAN	Primary Account Number
PIN	Personal Identification Number
POS	Point Of Sale
TNP-CG	TransNetPOS-CG
TNP-CG AX	TransNetPOS-CG ActiveX / OCX
VSDC	Visa Smart Debit Card
XML	Extensible Markup Language

Chapter 11: References

In addition to the resources referred to in the previous sections of this document, please visit the Partner Support Portal at:

http://help.precidia.com

The Partner Support Portal offers useful information such as sample code, how-to videos and applications that can viewed or downloaded.

Relevant Documentation

- "Using the TransNetPOS-CG to Access the XML API for POSLynx220 with TransNet"