GoEmerchant, LLC

Title: XML Gateway API Specification

PRODUCT: XML Gateway

Version 3.3.7

Document Control/Revisions

Document Modification	Version	Author/Editor	Date
Original Draft XML Gateway API with CIM	3.0.0	RJT	2/16/2009
Modified for formatting and clarity	3.0.1	MEB	2/17/2009
Added C#.NET sample code	3.0.2	MEB	2/20/2009
Added field transaction_center_id to supersede merchant field. Merchant field retained for legacy support.	3.0.3	MEB	2/24/2009
Added ACH support for batch upload in XML format, edit/delete CIM account.	3.0.4	MEB	3/17/2009
Updated ACH features to include proper reference to Debit/Credit and category text.	3.0.5	MEB	6/30/2009
Added ACH test account data	3.0.6	MEB	7/1/2009
Updated required fields for test account	3.0.7	MEB	7/7/2009
Added support for card validation and immediate void	3.0.8	MEB	7/9/2009
Added support for CIM_INSERT	3.0.9	RMB	7/23/2009
Updated QUERY operation documentation	3.1.0	RMB	12/22/2009
Added support for recurring billing start and end date	3.1.1	RMB	06/03/2010
Updated Docs for CIM_QUERY and is_debit_card	3.1.2	RJT	8/25/2010
Close Date and Additional MID / TID support	3.1.3	RJT	9/21/2010
Added section 6.1-6.2 with respect to the Sandbox testing	3.2.3	MEB	4/14/2011
Added new 'recurring_modify' operation type	3.2.4	RMB	4/20/2011
Added new 'void_partial' operation type and partial transaction support.	3.2.5	RJT	5/16/2011
Every other month recurring billing support	3.2.6	RJT	10/31/2011
Magensa Retail Support for MAG Data	3.3.0	RJT	11/17/2011
Query Operation Enhancement for trans_status flag	3.3.1	RJT	1/3/2012
Process ID Support	3.3.2	RJT	2/13/2012
Convenience Fee supported for select processors	3.3.3	RJT	3/28/2012
FutureX Encryption Support	3.3.4	RJT	10/18/12
Partial Reversal Support for MOTO/Retail	3.3.5	RJT	05/22/13
Added Posted By and Signature URL to Query result	3.3.6	JCH	08/20/13
Updated Format for ACH close_date	3.3.7	LJE	09/12/2013

TABLE OF CONTENTS

1. Introduction

The purpose of this document is to outline the technology and processes that drive the XML Gateway API. The API provides an interface into the gateway transaction processing network via secure sockets. The end user, after reading this document, should have sufficient knowledge based on the documentation and working samples provided to successfully integrate their ecommerce application into the XML Gateway API.

1.1. *Scope*

This specification will walk the reader through the each type of transaction supported by the XML gateway. A detailed description of each acceptable input message will be documented along with each possible response. This document will not cover language specific integrations, although there will be sample code provided in section 6.

1.2. *Product Description*

The XML gateway API is a programming interface that resides on transaction servers which communicate directly to credit card processing networks. The programming interface requires that the merchant, or their web programming staff, be sufficiently knowledgeable in programming skills in any programming or object oriented scripting language.

The XML Gateway API accepts the card purchasers information, including credit card information, billing address, total charge amount and order id and produces an authorization or decline directly from the merchant bank. The XML Gateway API provides methods to perform the following operations: AUTH, SALE, SETTLE, CREDIT, VOID, RETAIL SALE, RETAIL AUTH, QUERY, REAUTH, RESALE, CIM_AUTH, CIM_SALE, ACH_DEBIT, CIM_ACH_DEBIT, CIM_ACH_CREDIT, CIM_EDIT, CIM_DELETE, ACH_CREDIT, ACH_VOID and RETAIL_ALONE_CREDIT, RECURRING_MODIFY.

The information is passed via 128bit SSL https post in XML format. That post occurs in the background from the merchant's server. Thus the purchaser never leaves the merchant's website. The authorization information is returned in XML format with full error trapping and reporting to indicate the success or failure of the transaction.

URL for secure https post:
https://secure.goemerchant.com/secure/gateway/xmlgateway.aspx

Operational process:
1. The merchant's website produces the XML with all the required information depending on the operation type and performs an https (silent POST) to the gateway transaction servers.
2. The gateway's transaction servers determine the operation type and execute it.
3. Once the correct operation is carried out, the gateway assembles the response in XML.
4. The gateway's transactions servers send the XML response to the merchant's web or transaction servers indicating a success, failure or error.
5. The merchant parses the XML and notifies the customer or user of a successful request, failure or error.

1.3.   Intended Audience

This document is written exclusively for the use of approved merchants of GoEmerchant, LLC. It is intended to assist merchants with the task of integrating their ecommerce/retail solution with products offered or approved for integration with GoEmerchant's secure gateway processing networks.

2. Definitions

2.1.   Operation Types

Operation Type	Decription
AUTH	An AUTH operation is nothing more than a transaction that reserves the funds on a customer's credit card. The AUTH transaction must be accompanied by a SETTLE transaction in order for the merchant to receive funds.
SALE	A SALE operation produces a transaction that authorizes and captures the funds of the customer's credit card.
ACH_DEBIT	An ACH_DEBIT operation produces a transaction that debits a bank checking or savings account.
RETAIL_AUTH	A RETAIL_AUTH operation is nothing more than a transaction that reserves the funds on a customer's credit card. The RETAIL_AUTH transaction must be accompanied by a SETTLE transaction in order for the merchant to receive funds. Retail transactions require a retail merchant account.
RETAIL_SALE	A RETAIL_SALE operation produces a transaction that authorizes and captures the funds of the customer's credit card. Retail transactions require a retail merchant account.
REAUTH	A REAUTH operation uses the reference number from a previously successful AUTH/SALE/RETAIL_AUTH/RETAIL_SALE/CIM_AUTH/CIM_SALE transaction and initiates a new authorization using the billing information associated with the reference number provided. A follow up SETTLE transaction is required in order for the merchant to receive the funds for this authorization.
RESALE	A RESALE operation uses the reference number from a previously successful AUTH/SALE/RETAIL_AUTH/RETAIL_SALE/CIM_AUTH/CIM_SALE transaction and initiates a new authorization and capture using the billing information associated with the reference number provided.
CIM_AUTH	A CIM_AUTH operation uses the cim reference number from a previously entered customer record and initiates a new authorization using the billing information associated with the customer record identified. A follow up SETTLE transaction is required in order for the merchant to receive the funds for this authorization.
CIM_SALE	A CIM_SALE operation uses the cim reference number from a previously entered customer record and initiates a new authorization and capture using the billing information associated with the customer record identified.
CIM_ACH_DEBIT	A CIM_ACH_DEBIT operation uses the cim reference number from a previously entered customer record and initiates a new ach debit using the billing information associated with the customer record identified.
CIM_ACH_CREDIT	A CIM_ACH_CREDIT operation uses the cim reference number from a previously entered customer record and initiates a new ach credit using the billing information associated with the customer record identified.
CIM_INSERT	A CIM_INSERT operation allows a customer to be inserted into the CIM

	without first charging a customer.
CIM_EDIT	A CIM_EDIT operation allows access to a CIM stored record by cim_reference_number. Any of the account data stored with the CIM record can be modified such as card expiration.
CIM_DELETE	A CIM_DELETE operation will completely remove the associated CIM record from the system based on the cim_reference_number supplied.
CIM_QUERY	A CIM_QUERY operation will perform a query using the provided parameters and return the CIM record with all the sequence data associated with it for use in other CIM transactions.
CREDIT	CREDIT operations are performed against settled authorizations or sale transactions. Credits may only be performed against settled transactions and may not exceed the settled amount regardless of the original authorized amount. Multiple credits may be performed against a single settled authorization/sale up to the total settled amount.
ACH_CREDIT	ACH_CREDIT operations move money into a checking or savings account.
RETAIL_ALONE_CREDIT	A RETAIL_ALONE_CREDIT operation is a credit performed on new transactions that has never been processed through the gateway. This operation is different from a regular CREDIT, it does not issue credits against previously run transactions.
VOID	VOID operations are performed against AUTH/RETAIL_AUTH/CIM_AUTH transactions that have not settled. Voids will prevent the authorized transaction from ever settling.
ACH_VOID	ACH_VOID operations are performed against ACH transactions that have not posted. Voids will prevent the pending transaction from ever posting.
VOID_PARTIAL	VOID_PARTIAL operations are performed against all transactions that have not posted and are part of a partial transaction set. Voids will prevent the pending transaction from ever posting.
SETTLE	A SETTLE operation allows the merchant to submit the reference number of a corresponding AUTH/RETAIL_AUTH/CIM_AUTH/REAUTH transaction for batch settlement. The settlement occurs between 2:00 AM and 6:0 AM EST each day.
QUERY	QUERY operations allow the merchant to query the transaction database for all transaction types over a specified query range. The transaction data is returned with reference numbers in order for further operations to be performed against each transaction. NOTE – the full card number will not be exposed in a query transaction as per Visa/Mastercard requirements.
RECURRING_MODIFY	A RECURRING_MODIFY operation allows the merchant to insert a new recurring billing transaction from an existing authorization, or enable/disable/update an existing recurring billing transaction.

2.2.   XML Parameters/Field Definitions

Field Name	Field Type	Field Length	Description
Transaction Header Section			
merchant	Int	9	Unique identifier assigned by gateway. This is your unique Transaction Center number, not your 16 digit Merchant ID. **This field is no longer required and**

			exists for legacy support only.
transaction_center_id	Int	9	Unique identifier assigned by gateway. This is your unique Transaction Center number.
gateway_id	String	Uniqueidentifier	Unique identifier assigned by gateway. Can be found and or reset via the Options Tab in the Transaction Center.
operation_type	String	Varchar(20)	String specifying operation attempting to be run. Must be one of the supported operation types: AUTH, SALE, ACH_DEBIT, ACH_CREDIT, RETAIL_AUTH, RETAIL_SALE, CIM_AUTH, CIM_SALE, CIM_EDIT, CIM_DELETE, CIM_ACH_DEBIT, CIM_ACH_CREDIT, REAUTH, RESALE, CREDIT, RETAIL_ALONE_CREDIT, VOID, SETTLE, QUERY, RECURRING_MODIF
mid	String	Varchar(50)	Specific merchant number or firstfund username for the transaction to be processed under. This is only applicable if you have multiple accounts associated with your transaction_center_id.
tid	String	Varchar(50)	Specific terminal number this transaction. This is required if the mid is supplied and it is not an ACH transaction.
processor	String	Varchar(50)	Processor for the mid/tid combination suppled. Must be one of the following: echeck,paymentech,nova, fdc, vital, fifththird, fnms This is required if the mid is supplied.
processor_id	Numeric		Used in place of mid/tid/processor fields to identify the processor to run the transaction under. This is available in the transaction center.
Transaction Details Section			
order_id	String	Varchar(50)	Unique order id or invoice number. Cannot contain "insert", "update" or "delete".
Total	Numeric	(9,2)	Amount in US dollars. No dollar signs ($) or commas allowed.
Credit Card Data Section			
card_name	String	Varchar(30)	Type of card. Visa, mastercard, amex, discover etc. "no" or "yes" can also be used. If "no" or "yes" is used then the system will attempt to determine the card_name based on

			the card_number passed in.
card_number	Numeric	(19,0)	Credit card account number
card_exp	Numeric	(4,0)	Credit card expiration date. MMYY format.
cvv2	Numeric	(4,0)	Credit card security code, cvv2, cvc, cid
mag_data	String	Varchar (250)	Magnetic data containing at least the card number and card expiration. Obtained by swiping the credit card through a card swipe or VPOS unit.
cc_validate	Boolean	Bit	Set the value to 1 for doing small pre authorizations to determine a valid card, AVS or CVV2. The Transaction will be immediately voided. The field is NOT required for any other authorizations.
close_date	String	Varchar(8)	The date the transaction is intended to post. Format: MM/DD/YYYY
conv_fee	Numeric	(9,2)	Amount, already included in total, which was charged as a convenience fee. Not supported by all processors. Contact support to inquire if yours does.
Magensa Magnetic Data Encryption Section			
enctrack1	String		Encrypted Track1
enctrack2	String		Encrypted Track2
encmp	String		Encrypted MagnePrint Information
encstatus	String		Encrypted MagnePrint Status of Card Swipe
ksn	String		Serial number
encryptionblocktype	String		For the IPAD it will always be "2"
If using Magensa all of the above fields are required except for EncTrack1 and EncryptionBlockType. Those are optional. These fields can be ommitted completely if not being used.			
FutureX Magnetic Data Encryption Section			
enctrack1	String		Encrypted Track Data, can be either track
ksn	String		Serial number
encryptionblocktype	String		Slot number of the encryption key
swiper	String		Name of the swiper used to capture the data
If using FutureX all of the above fields are required except for EncryptionBlockType. It is recommended that you do pass the slot number for the decryption, but if you do not the primary production slot will be used.			
ACH Data Section			
aba	Numeric	(9, 0)	
dda	Numeric	(12, 0)	
ach_account_type	Char	(1)	C = Checking, S = Savings
ach_category_text	String	Varchar(23)	The category setup in the virtual that corresponds to this transaction.
close_date	String	Varchar(8)	The date the transaction is intended

			to post. Format: MM/DD/YYYY
ach_name	String	Varchar(75)	Account nickname Ex: CitiBank Checking
Cardholder/ACH owner Billing Address Section			
owner_name	String	Varchar (75)	Card holders/ACH account name
owner_street	String	Varchar (250)	Billing street address
owner_street2	String	Varchar (250)	Billing street address 2
owner_city	String	Varchar (100)	Billing city
owner_state	String	Varchar (100)	Billing state
owner_zip	String	Varchar (20)	Billing zipcode
owner_country	String	Varchar (200)	Billing country
owner_email	String	Varchar (300)	Billing email address
owner_phone	String	Varchar (25)	Billing phone number
Recurring Billing Section			
recurring	Boolean	Bit	0 or 1 – indicates if transaction is a recurring transaction. When this flag is turned on ("1") , a valid recurring_type must also accompany the transaction.
recurring_type	String	Varchar(12)	Indicates the recurring interval for this transaction to repeat. Will be ignored if the recurring flag is not "1". Accepted values: **daily, weekly, biweekly, monthly, bimonthly, quarterly, semiannually, annually**
recurring_start_date	String	Varchar(10)	The date the recurring transaction is intended to start. Format: MM/DD/YYYY
recurring_end_date	String	Varchar(10)	The date the recurring transaction is intended to end. Format: MM/DD/YYYY
send_invoice	Boolean	Bit	0 or 1 – indicates if the customer should receive an invoice each time a recurring billing transaction is processed.
is_ach	Boolean	Bit	0 or 1 - indicates whether the transaction being inserted/updated is an ACH transaction or not.
Partial Billing Section			
partial_id	Numeric	(20,0)	This numeric value is returned for partially approved transactions. It is used to tie subsequent transactions together to make up the larger transaction.
prevent_partial	Boolean	Bit	0 or 1 – indicates if a partial approval is allowed for this transaction. If this is set to 1 then a partially approved transaction will be rejected. This is 0 by default.
Customer IP Address Section			
remote_ip_address	String	Varchar(16)	Ip address of the customer contacting the merchant's site or

			application.
Purchase Card Level II Data Section			
purchase_card	Boolean	Bit	0 or 1 – indicates if the credit card used is a purchase card or not
customer_reference_number	String	Varchar(17)	Unique alpha-numeric value used to identify the card holder as a customer
local_tax_flag	Int	1	Flag set to 0, 1 or 2 that indicates the tax handling of the transaction being submitted. **0 - tax not provided** **1 - tax included** **2 - non-taxable transaction**
tax_amount	Numeric	(9,2)	Amount in US dollars. No dollar signs ($) or commas allowed.
Additional Fields Section			
total_additional_fields	Numeric	(4,0)	Indicates how many, if any, additional fields are associated with this transaction. This value specifies how many field_name/field_value pairs will be evaluated. Additional fields can be created in the transaction center for use with direct card authorizations and XML gateway transactions. You have to create the fields in the transaction center before attempting to pass them in for a gateway transaction.
field_name1..N	String	Varchar(100)	The <field_name> must match exactly the field name you used when creating the field in the transaction center.
field_value1..N	String	Varchar(100)	Value for the field specified in the field_name tag with the same numeric identifier.
Shipping Address Data Section			
shipping_name	String	Varchar(100)	Ship to recipient name
shipping_street	String	Varchar(250)	Ship to street address
shipping_street2	String	Varchar(250)	Ship to street address 2
shipping_city	String	Varchar(100)	Ship to city
shipping_state	String	Varchar(100)	Ship to state
shipping_zip	String	Varchar(20)	Ship to zipcode
shipping_country	String	Varchar(200)	Ship to country
shipping_phone	String	Varchar(25)	Ship to phone number
shipping_email	String	Varchar(300)	Ship to contact email
shipping_method	String	Varchar(100)	Shipping method used
Transaction Item Level III Data Section			
item_quantity1...1000	Numeric	(4,0)	Quantity of item purchased for the N index.
item_number1…1000	String	Varchar(200)	Item identifier, SKU code etc

item_description1…1000	String	Varchar(1000)	Text description of item
item_price1…1000	Numeric	(9,2)	Amount in US dollars. No dollar signs ($) or commas allowed.
CIM Data Section			
cim_ref_num	String	Varchar(75)	Unique alpha-numeric value used to identify the card holder as a customer. Can be used later for CIM specific operations.
is_retail	Boolean	Bit	0 or 1 indicating if the transaction should be submitted as a retail transaction.
cim_card_type	String	Varchar(30)	Card name such as Visa, Mastercard etc. This is used for CIM_AUTH/CIM_SALE operations and specifies which card to use in case the customer has multiple tied to their account.
card_sequence	Int	(2)	
ach_sequence	Int	(2)	
ship_sequence	Int	(2)	
Credit/Settle Data Section			
total_number_transactions	Numeric	(6,0)	Specifies the max N value for reference_number, credit_amount or settle_amount based on the operation type it is used with
reference_number1…n	Numeric	(12,0)	Unique transaction identifier returned by the gateway in the result of a previous transaction
credit_amount1…n	Numeric	(9,2)	Amount in US dollars. No dollar signs ($) or commas allowed.
settle_amount1…n	Numeric	(9,2)	Amount in US dollars. No dollar signs ($) or commas allowed.
Query Data Section			
trans_action	String	Varchar(30)	Type of transaction. Accepted values: ALL, CC, ACH.
card_type	String	Varchar(30)	Type of card. Visa, mastercard, amex, discover etc. used in query
trans_type	String	Varchar(12)	Transaction type to search for. Accepted values" sale, auth, void, credit, settle
trans_status	Int	(1,0)	2 – available for settlement 1 – successful transaction 0 – failed transaction
begin_date	String	Varchar(6)	MMDDYY format
begin_time	String	Varchar(6)	HHMMAM or HHMMPM
end_date	String	Varchar(6)	MMDDYY format
end_time	String	Varchar(6)	HHMMAM or HHMMPM
low_amount	Numeric	(9,2)	Amount in US dollars. No dollar signs ($) or commas allowed.
high_amount	Numeric	(9,2)	Amount in US dollars. No dollar signs ($) or commas allowed.

Additional Authorization Details Section			
retail			
reference_number			

3. API Specifications

3.1. Requirements for XMLGateway Utilization

1. Processing Platforms Supported.
    a. FDR (Omaha Platform)
    b. Elavon (Formerly Nova)
    c. Paymentech (Tampa Host Capture Platform)
    d. TSYS (formerly Vital/VisaNet)
    e. ACH (FirstFund, TSYS (FNMS))
2. Transaction Center ID
    a. Numeric access key assigned by GoEmerchant
    b. Unique to merchant
    c. Cannot be changed
    d. Required for API integration
3. Gateway ID
    a. Alpha-numeric passphrase assigned by GoEmerchant
    b. Can be changed via the gateway options area in the Transaction Center
    c. Should be changed every 90 days to ensure security
    d. Required for API integration
    e. Retrieved from the options tab in the Transaction Center
       https://secure.goemerchant.com/secure/transcenter/
4. MID / TID /Processor or Processor ID
    a. These parameters are only used for accounts that have multiple merchant accounts associated with a single transaction center id.
    b. If your account has multiple ACH accounts and you want to specify which one to process a transaction under, simply submit the mid parameter with the 3 character ACH username you have been assigned for that account.
    c. The values needed for each account can be found within your Transaction Center.
5. Secure URL
    https://secure.goemerchant.com/secure/gateway/xmlgateway.aspx
6. Development/Testing
    a. Test account is available
    b. Full test account information is documented in section 5

3.2. Auth/Sale Credit Card Operations - ACH Debit Operations

The AUTH/SALE Credit Card group of operations is the basic gateway function and the most commonly used function. These operation types are responsible for processing authorization only and sales on a customer's credit card.

Successful AUTH/SALE transactions will contain a reference_number in the generated response. This reference_number can be used to call upon this transcaction in future operations such as SETTLE or REAUTH.

Several variations of the AUTH/SALE operation type are available to be used. Each variation offers different features. Refer to their definitions below to decide which are most suitable for the integration desired. In addition, all authorization features can be combined into one message for example:
E-commerce Auth Only, with Level II Data, Recurring Billing "monthly", Level III Data, Additional Fields and CIM.

ACH Debit Operations will process a check transaction using the Automated Clearing House function. When an ACH Debit is initiated, the routing number is verified and the transaction is accepted. The ACH is assumed to be valid. The check will process on the specified date that is sent in the close_date field. All ACH transactions that are posted prior to 4pm Eastern time, will process that day and funds should be available the next morning.

ACH transactions may be done as a CIM transaction as well as and or in addition to a recurring billing transaction. All additional functions of credit card transactions, additional fields, recurring billing and Level III data are available for each transaction.

3.2.1. Auth/Sale Credit Card E-commerce

Fields utilized to attempt an auth and capture (SALE operation_type) for e-commerce.

Field Name	Field Value	Required
Transaction Header Section		
transaction_center_id	Number	X
gateway_id	Unique identifier	X
operation_type	"auth" or "sale"	X
mid	String	
tid	String	If mid
processor	String	If mid
processor_id	Numeric	Used in place of mid/tid/processor
Transaction Details Section		
order_id	String	X
total	Money	X
conv_fee	Money	
Credit Card Data Section		
card_name	String	X
card_number	Numeric	X
card_exp	Numeric	X
cvv2	numeric	
Card Holder Billing Address Section		
owner_name	String	X
owner_street	String	X
owner_street2	String	
owner_city	String	X
owner_state	String	X
owner_zip	String	X
owner_country	String	X
owner_email	String	
owner_phone	String	
Customer IP Address Section		
remote_ip_address	String	

3.2.1.1.   XML for E-commerce Auth/Sale

E-commerce (Auth and Capture) transaction

```xml
<?xml version="1.0" encoding="UTF-8"?>
<TRANSACTION>
<FIELDS>
<FIELD KEY="transaction_center_id">1264</FIELD>
<FIELD KEY="gateway_id">a91c38c3-7d7f-4d29-acc7-927b4dca0dbe</FIELD>
<FIELD KEY="operation_type">Sale </FIELD>
<FIELD KEY="order_id">Test Txn Oid</FIELD>
<FIELD KEY="total">1.00</FIELD>
<FIELD KEY="card_name">Visa</FIELD>
<FIELD KEY="card_number">4111111111111111</FIELD>
<FIELD KEY="card_exp">1018</FIELD>
<FIELD KEY="cvv2">123</FIELD>
<FIELD KEY="owner_name">Bob Tester</FIELD>
<FIELD KEY="owner_street">123 Test Rd.</FIELD>
<FIELD KEY="owner_street2"></FIELD>
<FIELD KEY="owner_city">Cityville</FIELD>
<FIELD KEY="owner_state">NJ</FIELD>
<FIELD KEY="owner_zip">08035</FIELD>
<FIELD KEY="owner_country">US</FIELD>
<FIELD KEY="owner_email"></FIELD>
<FIELD KEY="owner_phone">555-555-5555</FIELD>
<FIELD KEY="remote_ip_address">127.0.0.1</FIELD>
</FIELDS>
</TRANSACTION>
```

E-commerce Auth (Authorization only) transaction

```xml
<?xml version="1.0" encoding="UTF-8"?>
<TRANSACTION>
<FIELDS>
<FIELD KEY="transaction_center_id">1264</FIELD>
<FIELD KEY="gateway_id">a91c38c3-7d7f-4d29-acc7-927b4dca0dbe</FIELD>
<FIELD KEY="operation_type">Auth</FIELD>
<FIELD KEY="order_id">Test Txn Oid</FIELD>
<FIELD KEY="total">1.00</FIELD>
<FIELD KEY="card_name">Visa</FIELD>
<FIELD KEY="card_number">4111111111111111</FIELD>
<FIELD KEY="card_exp">1018</FIELD>
<FIELD KEY="cvv2">123</FIELD>
<FIELD KEY="owner_name">Bob Tester</FIELD>
<FIELD KEY="owner_street">123 Test Rd.</FIELD>
<FIELD KEY="owner_street2"></FIELD>
<FIELD KEY="owner_city">Cityville</FIELD>
<FIELD KEY="owner_state">NJ</FIELD>
<FIELD KEY="owner_zip">08035</FIELD>
<FIELD KEY="owner_country">US</FIELD>
<FIELD KEY="owner_email"></FIELD>
<FIELD KEY="owner_phone">555-555-5555</FIELD>
<FIELD KEY="remote_ip_address">127.0.0.1</FIELD>
```

```
</FIELDS>
</TRANSACTION>
```

3.2.2. Auth/Sale Credit Card Retail (Card Present)

Fields utilized to attempt an auth and capture (SALE operation_type) for retail.
To validate a credit card for AVS/CVV2, add the key cc_validate and set its value to 1. The cc_validate
function will authorize and then immediately void the charge.

NOTE – Retail Card Present "swiped" transactions never store Mag Data and CVV2/CVC/CID.

Field Name	Field Value	Required
Transaction Header Section		
transaction_center_id	Number	X
gateway_id	Unique identifier	X
operation_type	"retail_auth" or "retail_sale"	X
mid	String	
tid	String	If mid
processor	String	If mid
processor_id	Numeric	Used in place of mid/tid/processor
Transaction Details Section		
order_id	String	X
total	Money	X
conv_fee	Money	
Credit Card Data Section		
card_number	Numeric	X – required for e-comm & retail keyed
mag_data	String	X – required for retail Card Present if Magensa is not used
card_exp	Numeric	X – required for e-comm & retail keyed
cvv2	Numeric	
cc_validate	Bit	Optional 1=validate and void.
Magensa Magnetic Data Encryption Section		
enctrack1	string	
enctrack2	string	X – required if using Magensa
encmp	string	X – required if using Magensa
encstatus	string	X – required if using Magensa
ksn	string	X – required if using Magensa
encryptionblocktype	string	
FutureX Magnetic Data Encryption Section		
enctrack1	string	X – required if using FutureX

swiper	string	X – required if using FutureX
ksn	string	X – required if using FutureX
encryptionblocktype	string	
Card Holder Billing Address Section		
owner_name	String	
owner_street	String	
owner_street2	String	
owner_city	String	
owner_state	String	
owner_zip	String	
owner_country	String	
owner_email	String	
owner_phone	String	
Customer IP Address Section		
remote_ip_address	String	

3.2.2.1.   XML for Retail (Card Present) Auth/Sale

Retail (Auth and Capture) with Mag Data transaction transaction
```
<?xml version="1.0" encoding="UTF-8"?>
<TRANSACTION>
<FIELDS>
<FIELD KEY="transaction_center_id">1264</FIELD>
<FIELD KEY="gateway_id">a91c38c3-7d7f-4d29-acc7-927b4dca0dbe</FIELD>
<FIELD KEY="operation_type">retail_sale </FIELD>
<FIELD KEY="order_id">Test Txn Oid</FIELD>
<FIELD KEY="total">1.00</FIELD>
<FIELD KEY="mag_data">
%B4111111111111111^CardUser/John^101010100000019301000000877000000?;4111111111111111=0
305101193010877? </FIELD>
</FIELDS>
</TRANSACTION>
```

Retail Auth (Auth and Capture) with Keyed Card Data transaction
```
<?xml version="1.0" encoding="UTF-8"?>
<TRANSACTION>
<FIELDS>
<FIELD KEY="transaction_center_id">1264</FIELD>
<FIELD KEY="gateway_id">a91c38c3-7d7f-4d29-acc7-927b4dca0dbe</FIELD>
<FIELD KEY="operation_type">retail_sale </FIELD>
<FIELD KEY="order_id">Test Txn Oid</FIELD>
<FIELD KEY="total">1.00</FIELD>
<FIELD KEY="card_name">Visa</FIELD>
<FIELD KEY="card_number">4111111111111111</FIELD>
<FIELD KEY="card_exp">1018</FIELD>
<FIELD KEY="cvv2"> </FIELD>
</FIELDS>
</TRANSACTION>
```

Retail Auth (Authorization Only) with Mag Data transaction
<?xml version="1.0" encoding="UTF-8"?>
<TRANSACTION>
<FIELDS>
</FIELDS>
</TRANSACTION>

Retail Auth (Authorization Only) with Magensa transaction
<?xml version="1.0" encoding="UTF-8"?>
<TRANSACTION>
<FIELDS>
<FIELD KEY="EncTrack2">
1870AAD03014E23D6AA1E997F53717338BA6E331EF2BFA9C53CA467BF0FBA1D64C25CBAD2C2B2CBF</F
IELD>

</FIELDS>
</TRANSACTION>

Retail Auth (Authorization Only) with Keyed Card Data transaction
<?xml version="1.0" encoding="UTF-8"?>
<TRANSACTION>
<FIELDS>

```
</FIELDS>
</TRANSACTION>
```

3.2.3. ACH Debit

Fields utilized to attempt an ACH Debit (ACH_DEBIT operation_type).

Field Name	Field Value	Required
Transaction Header Section		
transaction_center_id	Number	X
gateway_id	Unique identifier	X
operation_type	"ach_debit"	X
mid	String	
Transaction Details Section		
order_id	String	X
total	Money	X
ACH Data Section (To Process ACH, the following 6 are required)		
aba	Numeric	X
dda	Numeric	X
ach_account_type	Char	X
ach_category_text	String	X
close_date	String	X (MM/DD/YYYY)
ach_name	String	X
Account Owner Data Section		
owner_name	String	X
owner_street	String	X
owner_street2	String	
owner_city	String	X
owner_state	String	X
owner_zip	String	X
owner_country	String	X
owner_email	String	
owner_phone	String	
Customer IP Address Section		
remote_ip_address	String	

3.2.3.1. XML for ACH Debit

ACH (Sale) transaction
```
<?xml version="1.0" encoding="UTF-8"?>
<TRANSACTION>
<FIELDS>
<FIELD KEY="transaction_center_id">1264</FIELD>
<FIELD KEY="gateway_id">a91c38c3-7d7f-4d29-acc7-927b4dca0dbe</FIELD>
<FIELD KEY="operation_type">ach_debit</FIELD>
<FIELD KEY="order_id">Test Txn Oid</FIELD>
<FIELD KEY="total">1.00</FIELD>
<FIELD KEY="aba">031200213</FIELD>
<FIELD KEY="dda">7596321546</FIELD>
<FIELD KEY="ach_account_type">C</FIELD>
```

```
<FIELD KEY="ach_category_text">Membership Fee</FIELD>
<FIELD KEY="close_date">06/23/2009</FIELD>
<FIELD KEY="ach_name">Citibank </FIELD>
<FIELD KEY="owner_name">Bob Tester</FIELD>
<FIELD KEY="owner_street">123 Test Rd.</FIELD>
<FIELD KEY="owner_street2"></FIELD>
<FIELD KEY="owner_city">Cityville</FIELD>
<FIELD KEY="owner_state">NJ</FIELD>
<FIELD KEY="owner_zip">08035</FIELD>
<FIELD KEY="owner_country">US</FIELD>
<FIELD KEY="owner_email"></FIELD>
<FIELD KEY="owner_phone">555-555-5555</FIELD>
<FIELD KEY="remote_ip_address">127.0.0.1</FIELD>
</FIELDS>
</TRANSACTION>
```

3.2.4.   Auth/Sale with Level II (B2B) Support

The Level II (B2B) Support Operation allows for additional fields to be submitted to process business to business card transactions. The merchant account must be setup to accept B2B transactions in order to use this operation type. The field types listed below will be included in a standard AUTH/SALE message.

Field Name	Field Value	Required
Purchase Card Level II Data Section		
purchase_card	1	X
customer_reference_number	String	X
local_tax_flag	Set group	X
shipping_zip	String	X
tax_amount	Money	X

3.2.4.1.   XML for Level II B2B transaction

```
<FIELD KEY="purchase_card">1</FIELD>
<FIELD KEY="customer_reference_number">1235493</FIELD>
<FIELD KEY="local_tax_flag">1</FIELD>
<FIELD KEY="shipping_zip">19036</FIELD>
<FIELD KEY="tax_amount">0.06</FIELD>
```

Example (Authorization Only) B2B e-commerce transaction
```
<?xml version="1.0" encoding="UTF-8"?>
<TRANSACTION>
<FIELDS>
<FIELD KEY="transaction_center_id">1264</FIELD>
<FIELD KEY="gateway_id">a91c38c3-7d7f-4d29-acc7-927b4dca0dbe</FIELD>
<FIELD KEY="operation_type">sale</FIELD>
<FIELD KEY="order_id">Test Txn Oid</FIELD>
<FIELD KEY="total">1.00</FIELD>
<FIELD KEY="card_name">Visa</FIELD>
<FIELD KEY="card_number">4111111111111111</FIELD>
<FIELD KEY="card_exp">1018</FIELD>
```

```
<FIELD KEY="purchase_card">1</FIELD>
<FIELD KEY="customer_reference_number">1235493</FIELD>
<FIELD KEY="local_tax_flag">1</FIELD>
<FIELD KEY="shipping_zip">19036</FIELD>
<FIELD KEY="tax_amount">0.06</FIELD>
<FIELD KEY="cvv2">123</FIELD>
<FIELD KEY="owner_name">Bob Tester</FIELD>
<FIELD KEY="owner_street">123 Test Rd.</FIELD>
<FIELD KEY="owner_street2"></FIELD>
<FIELD KEY="owner_city">Cityville</FIELD>
<FIELD KEY="owner_state">NJ</FIELD>
<FIELD KEY="owner_zip">08035</FIELD>
<FIELD KEY="owner_country">US</FIELD>
<FIELD KEY="owner_email"></FIELD>
<FIELD KEY="owner_phone">555-555-5555</FIELD>
<FIELD KEY="remote_ip_address">127.0.0.1</FIELD>
</FIELDS>
</TRANSACTION>
```

3.2.5. Auth/Sale with Additional Fields Support

Operation allows for additional fields to be submitted as part of a regular AUTH/SALE request.

<total_additional_fields> must contain the total number of additional fields passed in. It is the **N** value for the <field_name> and <field_value> tags.

Field Name	Field Value	Required
Additional Fields Section		
total_additional_fields	Numeric	X
field_name1…N	String	X
field_value1…N	Set group	X

3.2.5.1. XML for Additional Fields transaction

```
<FIELD KEY="total_additional_fields">3</FIELD>
<FIELD KEY="field_name1">color</FIELD>
<FIELD KEY="field_value1">red</FIELD>
<FIELD KEY="field_name2">size</FIELD>
<FIELD KEY="field_value2">medium</FIELD>
<FIELD KEY="field_name3">style</FIELD>
<FIELD KEY="field_value3">long sleeve</FIELD>
```

Example: Auth (Authorization Only) Additional Fields e-commerce transaction
```
<?xml version="1.0" encoding="UTF-8"?>
<TRANSACTION>
<FIELDS>
<FIELD KEY="transaction_center_id">1264</FIELD>
<FIELD KEY="gateway_id">a91c38c3-7d7f-4d29-acc7-927b4dca0dbe</FIELD>
<FIELD KEY="operation_type">auth </FIELD>
```

```
<FIELD KEY="order_id">Test Txn Oid</FIELD>
<FIELD KEY="total">1.00</FIELD>
<FIELD KEY="card_name">Visa</FIELD>
<FIELD KEY="card_number">4111111111111111</FIELD>
<FIELD KEY="card_exp">1018</FIELD>
<FIELD KEY="cvv2">123</FIELD>
<FIELD KEY="owner_name">Bob Tester</FIELD>
<FIELD KEY="owner_street">123 Test Rd.</FIELD>
<FIELD KEY="owner_street2"></FIELD>
<FIELD KEY="owner_city">Cityville</FIELD>
<FIELD KEY="owner_state">NJ</FIELD>
<FIELD KEY="owner_zip">08035</FIELD>
<FIELD KEY="owner_country">US</FIELD>
<FIELD KEY="owner_email"></FIELD>
<FIELD KEY="owner_phone">555-555-5555</FIELD>
<FIELD KEY="remote_ip_address">127.0.0.1</FIELD>
<FIELD KEY="total_additional_fields">3</FIELD>
<FIELD KEY="field_name1">color</FIELD>
<FIELD KEY="field_value1">red</FIELD>
<FIELD KEY="field_name2">size</FIELD>
<FIELD KEY="field_value2">medium</FIELD>
<FIELD KEY="field_name3">style</FIELD>
<FIELD KEY="field_value3">long sleeve</FIELD>
</FIELDS>
</TRANSACTION>
```

3.2.6.   Auth/Sale Level III Data Support

Level III Data Support is simply the ability to pass the data that supports level III transactions to the gateway for "storage" only.

The gateway DOES NOT support Level III transactions for interchange compliance.

Operation allows for all fields to be submitted that specify Level III data. Level III support must be enabled for the merchant account in order to use this operation type.

Level III data support is composed of two parts, item information and shipping information. One or both sets can be submitted with a Level III data support AUTH/SALE operation. Item information is limited to a maximum of 1000 line items per transaction.

Important Note: No calculations are performed on the line entry items. The item information is simply stored with the transaction. The amount specified in the <total> tag is what will be charged. Level III is not submitted to the processor for interchange compliance it is simply for reference.

Field Name	Field Value	Required
Transaction Item Level III Data Section		
item_quantity1...1000	Numeric	X
item_number1...1000	String	X
item_description1...1000	String	X

item_price1...1000	Money	X
Shipping Address Data Section		
shipping_name	String	X
shipping_street	String	
shipping_street2	String	
shipping_city	String	
shipping_state	String	
shipping_zip	String	
shipping_country	String	
shipping_phone	String	
shipping_email	String	
shipping_method	String	

3.2.6.1. XML Level III transaction

```
<FIELD KEY="item_quantity1">1</FIELD>
<FIELD KEY="item_price1">1.00</FIELD>
<FIELD KEY="item_number1">num 1</FIELD>
<FIELD KEY="item_description1">desc 1</FIELD>
<FIELD KEY="item_quantity2">2</FIELD>
<FIELD KEY="item_price2">2.00</FIELD>
<FIELD KEY="item_number2">num 2</FIELD>
<FIELD KEY="item_description2">desc 2</FIELD>
<FIELD KEY="item_quantityN">N</FIELD>
<FIELD KEY="item_priceN">N.NN</FIELD>
<FIELD KEY="item_numberN">num N</FIELD>
<FIELD KEY="item_descriptionN">desc N</FIELD>
<FIELD KEY="shipping_name">bob tester</FIELD>
<FIELD KEY="shipping_street">123 test rd</FIELD>
<FIELD KEY="shipping_street2"></FIELD>
<FIELD KEY="shipping_city ">here</FIELD>
<FIELD KEY="shipping_state">NJ </FIELD>
<FIELD KEY="shipping_zip">92835</FIELD>
<FIELD KEY="shipping_country">US</FIELD>
<FIELD KEY="shipping_phone ">999-999-9999</FIELD>
<FIELD KEY="shipping_email ">levelIII@email.com</FIELD>
<FIELD KEY="shipping_method ">UPS Ground</FIELD>
```

Example Auth (Authorization Only) Level III e-commerce transaction
```
<?xml version="1.0" encoding="UTF-8"?>
<TRANSACTION>
<FIELDS>
<FIELD KEY="transaction_center_id">1264</FIELD>
<FIELD KEY="gateway_id">a91c38c3-7d7f-4d29-acc7-927b4dca0dbe</FIELD>
<FIELD KEY="operation_type">sale </FIELD>
<FIELD KEY="order_id">Test Txn Oid</FIELD>
<FIELD KEY="total">1.00</FIELD>
<FIELD KEY="card_name">Visa</FIELD>
<FIELD KEY="card_number">4111111111111111</FIELD>
<FIELD KEY="card_exp">1018</FIELD>
<FIELD KEY="cvv2">123</FIELD>
```

```
<FIELD KEY="owner_name">Bob Tester</FIELD>
<FIELD KEY="owner_street">123 Test Rd.</FIELD>
<FIELD KEY="owner_street2"></FIELD>
<FIELD KEY="owner_city">Cityville</FIELD>
<FIELD KEY="owner_state">NJ</FIELD>
<FIELD KEY="owner_zip">08035</FIELD>
<FIELD KEY="owner_country">US</FIELD>
<FIELD KEY="owner_email"></FIELD>
<FIELD KEY="owner_phone">555-555-5555</FIELD>
<FIELD KEY="remote_ip_address">127.0.0.1</FIELD>
<FIELD KEY="item_quantity1">1</FIELD>
<FIELD KEY="item_price1">1.00</FIELD>
<FIELD KEY="item_number1">num 1</FIELD>
<FIELD KEY="item_description1">desc 1</FIELD>
<FIELD KEY="item_quantity2">2</FIELD>
<FIELD KEY="item_price2">2.00</FIELD>
<FIELD KEY="item_number2">num 2</FIELD>
<FIELD KEY="item_description2">desc 2</FIELD>
<FIELD KEY="item_quantity3">3</FIELD>
<FIELD KEY="item_price3">3.00</FIELD>
<FIELD KEY="item_number3">num 3</FIELD>
<FIELD KEY="item_description3">desc 3</FIELD>
<FIELD KEY="shipping_zip">92835</FIELD>
<FIELD KEY="shipping_name">bob tester</FIELD>
<FIELD KEY="shipping_street">123 test rd</FIELD>
 </FIELDS>
</TRANSACTION>
```

3.2.7.   Auth/Sale/ACH Debit/Credit CIM (Customer Information Management) Support

CIM is short for Customer Information Management. CIM Operations allow additional fields to be submitted that specify a transaction to be stored permanently in the CIM database. When a transaction is submitted and stored in the CIM database, additional sales/authorizations may be submitted without the need for card holder/ach check data. CIM support must be enabled for the merchant account in order to use this operation type.

NOTE – Retail card present "swiped" transactions never store Mag Data and CVV2/CVC/CID. Only the card number and expiration will be retained for a future authorization peformed against the CIM record.

A CIM can have 0 or more shipping addresses associated with their record. The merchant can pass in the specified shipping fields to create or update the specified shipping address for the customer record or they can be omitted to not create or update any shipping information for the customer tied to the <cim_ref_num>.

The CIM spec provides a method to send in a new transaction or perform a Credit Card Auth/Sale or and ACH Debit/Credit transaction against an existing record.

Important Note: The <cim_ref_num> should be unique to the customer being billed. If the number is not unique it will cause an update to occur for the customer tied to that <cim_ref_num> already. It is intended to work this way to enable customer updates to occur in one step when a new transaction is processed for their account.

CIM Field Requirements for (New) Customer Transaction

Field Name	Field Value	Required
Transaction Header Section		
transaction_center_id	Number	X
gateway_id	Unique identifier	X
Operation_type	"auth", "sale" or "ach_debit", "ach_credit"	X
mid	String	
tid	String	If mid and not ACH
processor	String	If mid and not ACH
processor_id	Numeric	Used in place of mid/tid/processor
Transaction Details Section		
order_id	String	X
Total	Money	X
CIM Details Section		
cim_ref_num	String	X
Shipping Address Section		
shipping_name	String	
shipping_street	String	
shipping_street2	String	
shipping_city	String	
shipping_state	String	
shipping_zip	String	
shipping_country	String	
shipping_phone	String	
shipping_email	String	
shipping_method	String	
Credit Card Data Section		
card_name	String	X
card_number	Numeric	X – required for e-comm & retail keyed
mag_data	String	X – required for retail Card Present
card_exp	Numeric	X – required for e-comm & retail keyed
cvv2	Numeric	
ACH Data Section		
aba	Numeric	X- required for ACH
dda	Numeric	X- required for ACH
ach_account_type	Char	X- required for ACH
ach_category_text	String	X- required for ACH
close_date	String	X- required for ACH (MM/DD/YYYY)
ach_name	String	X- required for ACH
Card Holder/ACH Owner Billing Address Section		
owner_name	String	X
owner_street	String	X
owner_street2	String	
owner_city	String	X
owner_state	String	X
owner_zip	String	X

owner_country	String	X
owner_email	String	
owner_phone	String	
Customer IP Address Section		
remote_ip_address	String	

CIM Field Requirements for (Existing) Customer Transaction

Field Name	Field Value	Required
Transaction Header Section		
transaction_center_id	Number	X
Gateway_id	Unique identifier	X
operation_type	"cim_auth", "cim_sale" or "cim_ach_debit", "cim_ach_credit"	X
mid	String	
tid	String	If mid and not ACH
processor	String	If mid and not ACH
processor_id	Numeric	Used in place of mid/tid/processor
Transaction Details Section		
order_id	String	X
Total	Money	X
CIM Data Section		
cim_ref_num	String	X – if no reference_number
is_retail	Boolean	
cim_card_type	String	
card_sequence	Int	X – Required if multiple card types stored.
ach_sequence	Int	X – Required if multiple accounts stored.
ship_sequence	Int	X – Required if multiple shipping addresses stored.

3.2.7.1. XML E-commerce Auth/Sale CIM (New Customer) transaction

Example: Sale (Auth and Capture) CIM new customer e-commerce transaction
```
<?xml version="1.0" encoding="UTF-8"?>
<TRANSACTION>
<FIELDS>
<FIELD KEY="transaction_center_id">1264</FIELD>
<FIELD KEY="gateway_id">a91c38c3-7d7f-4d29-acc7-927b4dca0dbe</FIELD>
<FIELD KEY="operation_type">auth</FIELD>
<FIELD KEY="order_id">Test Txn Oid</FIELD>
<FIELD KEY="total">1.00</FIELD>
<FIELD KEY="card_name">Visa</FIELD>
<FIELD KEY="card_number">4111111111111111</FIELD>
<FIELD KEY="card_exp">1018</FIELD>
<FIELD KEY="cvv2">123</FIELD>
<FIELD KEY="owner_name">Bob Tester</FIELD>
```

```
<FIELD KEY="owner_street">123 Test Rd.</FIELD>
<FIELD KEY="owner_street2"></FIELD>
<FIELD KEY="owner_city">Cityville</FIELD>
<FIELD KEY="owner_state">NJ</FIELD>
<FIELD KEY="owner_zip">08035</FIELD>
<FIELD KEY="owner_country">US</FIELD>
<FIELD KEY="owner_email"></FIELD>
<FIELD KEY="owner_phone">555-555-5555</FIELD>
<FIELD KEY="remote_ip_address">127.0.0.1</FIELD>
<FIELD KEY="cim_ref_num">100231</FIELD>
<FIELD KEY="shipping_zip">92835</FIELD>
<FIELD KEY="shipping_name">bob tester</FIELD>
<FIELD KEY="shipping_street">123 test rd</FIELD>
 </FIELDS>
</TRANSACTION>
```

3.2.7.2.   XML E-commerce Auth/Sale CIM (Existing Customer) transaction

Example: Sale (Auth and Capture) CIM existing customer e-commerce transaction
```
<?xml version="1.0" encoding="UTF-8"?>
<TRANSACTION>
<FIELDS>
<FIELD KEY="transaction_center_id">1264</FIELD>
<FIELD KEY="gateway_id">a91c38c3-7d7f-4d29-acc7-927b4dca0dbe</FIELD>
<FIELD KEY="operation_type">cim_sale</FIELD>
<FIELD KEY="order_id">Test Txn Oid</FIELD>
<FIELD KEY="total">1.00</FIELD>
<FIELD KEY="cim_ref_num">10015</FIELD>
<FIELD KEY="cim_card_type">visa</FIELD>
<FIELD KEY="card_sequence">1</FIELD>
<FIELD KEY="is_retail">0</FIELD>
</FIELDS>
</TRANSACTION>
```

3.2.7.3.   XML Retail Auth/Sale CIM (Existing Customer) transaction

Example: Sale (Auth and Capture) CIM existing customer retail transaction
```
<?xml version="1.0" encoding="UTF-8"?>
<TRANSACTION>
<FIELDS>
<FIELD KEY="transaction_center_id">1264</FIELD>
<FIELD KEY="gateway_id">a91c38c3-7d7f-4d29-acc7-927b4dca0dbe</FIELD>
<FIELD KEY="operation_type">cim_sale</FIELD>
<FIELD KEY="order_id">Test Txn Oid</FIELD>
<FIELD KEY="total">1.00</FIELD>
<FIELD KEY="cim_ref_num">10015</FIELD>
<FIELD KEY="cim_card_type">visa</FIELD>
<FIELD KEY="card_sequence">1</FIELD>
<FIELD KEY="is_retail">1</FIELD>
</FIELDS>
```

```
</TRANSACTION>
```

3.2.7.4. XML ACH Debit CIM (New Customer) transaction

Example: ACH Debit CIM new customer ACH transaction
```
<?xml version="1.0" encoding="UTF-8"?>
<TRANSACTION>
<FIELDS>
<FIELD KEY="transaction_center_id">1264</FIELD>
<FIELD KEY="gateway_id">a91c38c3-7d7f-4d29-acc7-927b4dca0dbe</FIELD>
<FIELD KEY="operation_type">cim_ach_debit</FIELD>
<FIELD KEY="order_id">Test Txn Oid</FIELD>
<FIELD KEY="total">1.00</FIELD>
<FIELD KEY="aba">031253032 </FIELD>
<FIELD KEY="dda">7869453216 </FIELD>
<FIELD KEY="ach_type">C </FIELD>
<FIELD KEY="ach_category_text">Membership Fee</FIELD>
<FIELD KEY="close_date">06/23/2009</FIELD>
<FIELD KEY="ach_name">Citibank</FIELD>
<FIELD KEY="owner_name">Bob Tester</FIELD>
<FIELD KEY="owner_street">123 Test Rd.</FIELD>
<FIELD KEY="owner_street2"></FIELD>
<FIELD KEY="owner_city">Cityville</FIELD>
<FIELD KEY="owner_state">NJ</FIELD>
<FIELD KEY="owner_zip">08035</FIELD>
<FIELD KEY="owner_country">US</FIELD>
<FIELD KEY="owner_email"></FIELD>
<FIELD KEY="owner_phone">555-555-5555</FIELD>
<FIELD KEY="remote_ip_address">127.0.0.1</FIELD>
<FIELD KEY="cim_ref_num">100231</FIELD>
<FIELD KEY="shipping_zip">92835</FIELD>
<FIELD KEY="shipping_name">bob tester</FIELD>
<FIELD KEY="shipping_street">123 test rd</FIELD>
 </FIELDS>
</TRANSACTION>
```

3.2.7.5. XML ACH Debit CIM (Existing Customer) transaction

Example: ACH Debit CIM existing customer ACH transaction
```
<?xml version="1.0" encoding="UTF-8"?>
<TRANSACTION>
<FIELDS>
<FIELD KEY="transaction_center_id">1264</FIELD>
<FIELD KEY="gateway_id">a91c38c3-7d7f-4d29-acc7-927b4dca0dbe</FIELD>
<FIELD KEY="operation_type">cim_ach_debit </FIELD>
<FIELD KEY="order_id">Test Txn Oid</FIELD>
<FIELD KEY="total">1.00</FIELD>
<FIELD KEY="cim_ref_num">10015</FIELD>
<FIELD KEY="ach_sequence">1</FIELD>
</FIELDS>
```

```
</TRANSACTION>
```

3.2.8.   Auth/Sale Recurring Billing Support

The Recurring Billing option will attempt a sale or auth only transaction for e-commerce or retail and mark the transaction as a recurring payment. Recurring payment options are:
1. Daily
2. Weekly
3. Biweekly
4. Monthly
5. Bimonthly (Every other month)
6. Quarterly
7. Semiannually
8. Annually

Recurring Billing Field Requirements.

Field Name	Field Value	Required
Recurring Billing Data Section		
recurring	1	X
recurring_type	daily, weekly, biweekly, monthly, bimonthly, quarterly, semiannually, annually	X
recurring_start_date	MM/DD/YYYY	
recurring_end_date	MM/DD/YYYY	

3.2.8.1.   XML E-commerce Auth/Sale Recurring Billing transaction

```
<FIELD KEY="recurring">1</FIELD>
<FIELD KEY="recurring_type">monthly</FIELD>
<FIELD KEY="recurring_start_date">01/01/2012</FIELD>
<FIELD KEY="recurring_type">12/31/2015</FIELD>
```

Example: Sale (Authorization Only) Recurring Billing e-commerce transaction
```
<?xml version="1.0" encoding="UTF-8"?>
<TRANSACTION>
<FIELDS>
<FIELD KEY="transaction_center_id">1264</FIELD>
<FIELD KEY="gateway_id">a91c38c3-7d7f-4d29-acc7-927b4dca0dbe</FIELD>
<FIELD KEY="operation_type">auth</FIELD>
<FIELD KEY="order_id">Test Txn Oid</FIELD>
<FIELD KEY="total">1.00</FIELD>
<FIELD KEY="card_name">Visa</FIELD>
<FIELD KEY="card_number">4111111111111111</FIELD>
<FIELD KEY="card_exp">1018</FIELD>
<FIELD KEY="cvv2">123</FIELD>
<FIELD KEY="owner_name">Bob Tester</FIELD>
```

```
<FIELD KEY="owner_street">123 Test Rd.</FIELD>
<FIELD KEY="owner_street2"></FIELD>
<FIELD KEY="owner_city">Cityville</FIELD>
<FIELD KEY="owner_state">NJ</FIELD>
<FIELD KEY="owner_zip">08035</FIELD>
<FIELD KEY="owner_country">US</FIELD>
<FIELD KEY="owner_email"></FIELD>
<FIELD KEY="owner_phone">555-555-5555</FIELD>
<FIELD KEY="remote_ip_address">127.0.0.1</FIELD>
<FIELD KEY="recurring">1</FIELD>
<FIELD KEY="recurring_type">monthly</FIELD>
</FIELDS>
</TRANSACTION>
```

3.2.9. Auth/Sale Partial Billing Support

It is possible that a transaction only partially approves. When this occurs, the response will contain the following keys: **is_partial, partial_id, partial_amount,** and **original_full_amount**. Your program will use these values to determine the remaining balance the customer owes and allow you to ask for an additional form of payment. In order to tie these transactions together you can pass the **partial_id** parameter that this transaction set was assigned in subsequent payments. It is possible that subsequent payments will also only partially approve. In this case they will not be assigned a new **partial_id**, but the **is_partial** flag will be set along with the amounts for that transaction.

Partial Billing Field Requirements.

Field Name	Field Value	Required
Partial Billing Data Section		
partial_id	Numeric	X – value was returned in pervious auth that wasn't fully processed

3.2.9.1.   XML E-commerce Auth/Sale Partial Billing transaction

```
<FIELD KEY="partial_id">12365841</FIELD>
```

Example: Sale (Authorization Only) Partial Billing e-commerce transaction
```
<?xml version="1.0" encoding="UTF-8"?>
<TRANSACTION>
<FIELDS>
<FIELD KEY="transaction_center_id">1264</FIELD>
<FIELD KEY="gateway_id">a91c38c3-7d7f-4d29-acc7-927b4dca0dbe</FIELD>
<FIELD KEY="operation_type">auth</FIELD>
<FIELD KEY="order_id">Test Txn Oid</FIELD>
<FIELD KEY="total">10.00</FIELD>
<FIELD KEY="card_name">Visa</FIELD>
<FIELD KEY="card_number">4111111111111111</FIELD>
<FIELD KEY="card_exp">1018</FIELD>
<FIELD KEY="cvv2">123</FIELD>
```

```
<FIELD KEY="owner_name">Bob Tester</FIELD>
<FIELD KEY="owner_street">123 Test Rd.</FIELD>
<FIELD KEY="owner_street2"></FIELD>
<FIELD KEY="owner_city">Cityville</FIELD>
<FIELD KEY="owner_state">NJ</FIELD>
<FIELD KEY="owner_zip">08035</FIELD>
<FIELD KEY="owner_country">US</FIELD>
<FIELD KEY="owner_email"></FIELD>
<FIELD KEY="owner_phone">555-555-5555</FIELD>
<FIELD KEY="remote_ip_address">127.0.0.1</FIELD>
<FIELD KEY="partial_id"> 12365841</FIELD>
</FIELDS>
</TRANSACTION>
```

The above transaction would be a second transaction tied to a previously run transaction that only partially approved and did not cover the last $10.00 of the transaction.

3.2.10. Re-Authorize/Re-Sale/Re-Debit Operations

This will attempt to authorize and capture (RESALE operation type) the transaction entered based on a <reference_number> of a previously authorized transaction without the need to re-input card data or ach check data.

Re-Authorize/Re-SaleRe-Debit Operations

Field Name	Field Value	Required
Transaction Header Section		
transaction_center_id	Number	X
gateway_id	Unique identifier	X
operation_type	"reauth", "resale", "redebit"	X
mid	String	
tid	String	If mid
processor	String	If mid
processor_id	Numeric	Used in place of mid/tid/processor
Transaction Details Section		
order_id	String	X
total	Money	X
Additional Authorization Data Section		
reference_number	Numeric	X
is_ach	Boolean	X – If ACH
Recurring Billing Data Section		
recurring	Boolean	
recurring_type	Set group	
recurring_start_date	String	
recurring_end_date	String	

3.2.10.1. XML E-commerce ReAuth/ReSale existing transaction

Reference Number RESALE e-commerce transaction:
```
<?xml version="1.0" encoding="UTF-8"?>
<TRANSACTION>
<FIELDS>
<FIELD KEY="transaction_center_id">1264</FIELD>
<FIELD KEY="gateway_id">a91c38c3-7d7f-4d29-acc7-927b4dca0dbe</FIELD>
<FIELD KEY="operation_type">resale</FIELD>
<FIELD KEY="order_id">Test Txn Oid</FIELD>
<FIELD KEY="total">1.00</FIELD>
<FIELD KEY="reference_number">125095</FIELD>
</FIELDS>
</TRANSACTION>
```

Reference Number REAUTH (Authorize Only) e-commerce transaction:
```
<?xml version="1.0" encoding="UTF-8"?>
<TRANSACTION>
<FIELDS>
<FIELD KEY="transaction_center_id">1264</FIELD>
<FIELD KEY="gateway_id">a91c38c3-7d7f-4d29-acc7-927b4dca0dbe</FIELD>
<FIELD KEY="operation_type">reauth</FIELD>
<FIELD KEY="order_id">Test Txn Oid</FIELD>
<FIELD KEY="total">1.00</FIELD>
<FIELD KEY="reference_number">125095</FIELD>
</FIELDS>
</TRANSACTION>
```

3.2.11. Auth/Sale Responses

Response for Credit Card/ACH Transactions (Non-CIM)
```
<?xml version="1.0" encoding="UTF-8"?>
<RESPONSE>
<FIELDS>
<FIELD KEY="status">0-error  1-success  2-declined</FIELD>
<FIELD KEY="auth_code">character code sent by the bank </FIELD>
<FIELD KEY="auth_response">message from the bank </FIELD>
<FIELD KEY="avs_code">avs code from the bank</FIELD>
<FIELD KEY="cvv2_code">cvv2 code from the bank</FIELD>
<FIELD KEY="order_id">echoed back from original post</FIELD>
<FIELD KEY="reference_number">returned for use with credits/voids/settles</FIELD>
<FIELD KEY="available_balance">balance left on card if available. Used in Gift cards most often</FIELD>
<FIELD KEY="error">error text</FIELD>
</FIELDS>
</RESPONSE>
```

Response for Credit Card/ACH Transactions (CIM)
```
<?xml version="1.0" encoding="UTF-8"?>
<RESPONSE>
<FIELDS>
<FIELD KEY="status">0-error  1-success  2-declined</FIELD>
<FIELD KEY="auth_code">character code sent by the bank </FIELD>
<FIELD KEY="auth_response">message from the bank </FIELD>
```

```
<FIELD KEY="avs_code">avs code from the bank</FIELD>
<FIELD KEY="cvv2_code">cvv2 code from the bank</FIELD>
<FIELD KEY="order_id">echoed back from original post</FIELD>
<FIELD KEY="reference_number">returned for use with credits/voids/settles</FIELD>
<FIELD KEY="error">error text</FIELD>
<FIELD KEY="card_sequence">echoed back sequence number of the card input</FIELD>
<FIELD KEY="ach_sequence">echoed back sequence number of the ach (check) input</FIELD>
<FIELD KEY="ship _sequence">echoed back sequence number of the shipping address input</FIELD>
<FIELD KEY="available_balance">balance left on card if available. Used in Gift cards most often</FIELD>
</FIELDS>
</RESPONSE>
```

Response for Partail Credit Card/ACH Transactions (CIM)
```
<FIELD KEY="is_partial">1</FIELD>
<FIELD KEY="partial_id">numeric value identifying partial transaction set.</FIELD>
<FIELD KEY="original_full_amount">auth amount originally submitted for this current
transaction.</FIELD>
<FIELD KEY="partial_amount">amount  actually billed</FIELD>
```

STATUS – 0 - response:
```
<FIELD KEY="status">0</FIELD>
<FIELD KEY="auth_code"></FIELD>
<FIELD KEY="auth_response"></FIELD>
<FIELD KEY="avs_code"></FIELD>
<FIELD KEY="cvv2_code"></FIELD>
<FIELD KEY="order_id">ECHO Order ID Passed in</FIELD>
<FIELD KEY="reference_number"></FIELD>
 <FIELD KEY="error">DESCRIPTIVE ERROR MESSAGE</FIELD>
```

Notice most fields are blank when status = 0! If the status is a 1 or 2 then the fields will be filled in with the appropriate responses

ERROR –
As is commonplace in XML, fields with no values can be self closing, if no error is returned in the error field, then it will return as <FIELD KEY="error" />. Please note that some successful transactions will return the error field empty with the self closing syntax.

If status is 1 the response will look like this:
```
<FIELD KEY="status">1</FIELD>
<FIELD KEY="auth_code">AB5943</FIELD>
<FIELD KEY="auth_response">Approved</FIELD>
<FIELD KEY="avs_code">Y</FIELD>
<FIELD KEY="cvv2_code">U</FIELD>
<FIELD KEY="order_id">Test Txn Oid</FIELD>
<FIELD KEY="reference_number">1549843</FIELD>
<FIELD KEY="error" />
```

reference_number – this is the number used in later operations to reference specific transactions for credits, voids and settles.
partial_id – this is used in later operations to identify the partial transaction set the new transaction being submitted is part of.

3.3.   *Credit Operations*

CREDIT operations are performed against settled authorizations or sale transactions for Credit Cards. ACH transactions perform Credits to move money into a specified bank checking or savings account. Credit Card e-commerce and MO/TO account credits may only be performed against settled transactions and may not exceed the settled amount regardless of the original authorized amount. Multiple credits may be performed against a single settled authorization/sale. Retail Credit Card transactions may perform a credit with a matching authorization and settle record. ACH Credit transactions may be performed at any time to move money into an existing bank account such as for payroll direct deposit.

3.3.1.   E-commerce/MOTO/Retail (Card Present) Credit Existing Transaction

The <total_number_transactions> field specifies the **N** number of <reference_number> and <credit_amount> tags to look for. The total_number_transactions field must match the number of transactions entered or the gateway will respond with an error.

Ex: total_number_transactions = 5 There must be reference_numbers1..5 etc.

Field Name	Field Value	Required
Transaction Header Section		
transaction_center_id	Number	X
gateway_id	Unique identifier	X
operation_type	"credit"	X
mid	String	
tid	String	If mid
processor	String	If mid
processor_id	Numeric	Used in place of mid/tid/processor
Credit/Settle Data Section		
total_number_transactions	String	X
reference_number1…n	numeric	X
credit_amount1…n	Money	X
conv_fee1…n	Money	
Customer IP Address Section		
remote_ip_address	String	

```
<?xml version="1.0" encoding="UTF-8"?>
<TRANSACTION>
<FIELDS>
<FIELD KEY="transaction_center_id">1264</FIELD>
<FIELD KEY="gateway_id">a91c38c3-7d7f-4d29-acc7-927b4dca0dbe</FIELD>
<FIELD KEY="operation_type">credit</FIELD>
<FIELD KEY="total_number_transactions">2</FIELD>
<FIELD KEY="reference_number1">1222</FIELD>
<FIELD KEY="reference_number2">33334</FIELD>
<FIELD KEY="credit_amount1">5.00</FIELD>
<FIELD KEY="credit_amount2">1.00</FIELD>
<FIELD KEY="remote_ip_address">127.0.0.1</FIELD>
</FIELDS>
</TRANSACTION>
```

3.3.2. Retail (Card Present) Single Transaction Credit

The Retail Alone Credit operation allows the merchant to credit a transaction that has NOT been previously run through the gateway. This must be enabled on the merchant's account before attempting.

Field Name	Field Value	Required
Transaction Header Section		
transaction_center_id	Number	X
gateway_id	Unique identifier	X
operation_type	"retail_alone_credit"	X
mid	String	
tid	String	If mid
processor	String	If mid
processor_id	Numeric	Used in place of mid/tid/processor
Transaction Details Section		
order_id	String	X
Total	Money	X
Credit Card Data Section		
mag_data	String	X – Required for Card Present Swipe Transaction.
card_name	String	X – Required if mag data blank
card_number	Numeric	X – Required if mag data blank.
card_exp	Numeric	X – Required if mag data blank.
cvv2	Numeric	
Card Holder Billing Address Section		
owner_name	String	
owner_street	String	
owner_street2	String	
owner_city	String	
owner_state	String	
owner_zip	String	
owner_country	String	
owner_email	String	
owner_phone	String	
Customer IP Address Section		
remote_ip_address	String	

Example: Retail Card Present (Single Credit) with Mag Data transaction
<?xml version="1.0" encoding="UTF-8"?>
<TRANSACTION>
<FIELDS>

```
<FIELD KEY="mag_data">
%B4111111111111111^CardUser/John^1010101000000193010000000877000000?;4111111111111111=0
305101193010877? </FIELD>
</FIELDS>
</TRANSACTION>
```

Example: Retail Card Present (Single Credit) with keyed card data transaction
```
<?xml version="1.0" encoding="UTF-8"?>
<TRANSACTION>
<FIELDS>
<FIELD KEY="transaction_center_id">1264</FIELD>
<FIELD KEY="gateway_id">a91c38c3-7d7f-4d29-acc7-927b4dca0dbe</FIELD>
<FIELD KEY="operation_type">retail_alone_credit</FIELD>
<FIELD KEY="order_id">Test Txn Oid</FIELD>
<FIELD KEY="total">1.00</FIELD>
<FIELD KEY="card_name">Visa</FIELD>
<FIELD KEY="card_number">4111111111111111</FIELD>
<FIELD KEY="card_exp">1018</FIELD>
<FIELD KEY="cvv2"> </FIELD>
</FIELDS>
</TRANSACTION>
```

3.3.3. ACH Credit

Fields utilized to attempt an ACH Credit (ACH_CREDIT operation_type).

Field Name	Field Value	Required
Transaction Header Section		
transaction_center_id	Number	X
gateway_id	Unique identifier	X
operation_type	"ach_credit"	X
mid	String	
processor	String	If mid
processor_id	Numeric	Used in place of mid/tid/processor
Transaction Details Section		
order_id	String	X
total	Money	X
ACH Data Section		
aba	Numeric	X
dda	Numeric	X
ach_account_type	Char	X
ach_category_text	String	X
close_date	String	X (MM/DD/YYYY)
ach_name	String	X
Account Owner Data Section		
owner_name	String	X
owner_street	String	X
owner_street2	String	
owner_city	String	X
owner_state	String	X

owner_zip	String	X
owner_country	String	X
owner_email	String	
owner_phone	String	
Customer IP Address Section		
remote_ip_address	String	

3.3.3.1. XML for ACH Credit

ACH (Sale) transaction
```
<?xml version="1.0" encoding="UTF-8"?>
<TRANSACTION>
<FIELDS>
<FIELD KEY="transaction_center_id">1264</FIELD>
<FIELD KEY="gateway_id">a91c38c3-7d7f-4d29-acc7-927b4dca0dbe</FIELD>
<FIELD KEY="operation_type">ach_credit</FIELD>
<FIELD KEY="order_id">Test Txn Oid</FIELD>
<FIELD KEY="total">1.00</FIELD>
<FIELD KEY="aba">031200213</FIELD>
<FIELD KEY="dda">7596321546</FIELD>
<FIELD KEY="ach_account_type">C</FIELD>
<FIELD KEY="ach_category_text">Membership Fee</FIELD>
<FIELD KEY="close_date">06/23/2009</FIELD>
<FIELD KEY="ach_name">Citibank</FIELD>
<FIELD KEY="owner_name">Bob Tester</FIELD>
<FIELD KEY="owner_street">123 Test Rd.</FIELD>
<FIELD KEY="owner_street2"></FIELD>
<FIELD KEY="owner_city">Cityville</FIELD>
<FIELD KEY="owner_state">NJ</FIELD>
<FIELD KEY="owner_zip">08035</FIELD>
<FIELD KEY="owner_country">US</FIELD>
<FIELD KEY="owner_email"></FIELD>
<FIELD KEY="owner_phone">555-555-5555</FIELD>
<FIELD KEY="remote_ip_address">127.0.0.1</FIELD>
</FIELDS>
</TRANSACTION>
```

3.3.4. E-commerce/MOTO/Retail (Card Present) Credit Existing Transaction Response

Returned from XMLGateway for operation_type credit:
```
<?xml version="1.0" encoding="UTF-8"?>
<RESPONSE>
  <FIELDS>
    <FIELD KEY="total_transactions_credited">Total number of transactions credited</FIELD>
    <FIELD KEY="status1..n"> 0-error  1-success  2-Rejected</FIELD>
    <FIELD KEY="response1..n">Response text returned by Transaction Center</FIELD>
    <FIELD KEY="reference_number1..n">reference number of credited transaction</FIELD>
    <FIELD KEY="credit_amount1..n">amount credited</FIELD>
    <FIELD KEY="error1..n">error text</FIELD>
  </FIELDS>
```

</RESPONSE>

NOTE

total_transactions_credited - the number of successful credits performed on the list of credits attempted. This value can be used to check the successful statuses of the returned credits and make sure the number of successful statuses is the same as the total_transactions_credited.

1..n – this means that if the total_number_transactions = 3 which was passed in then there should be a status, response, reference_number, credit_amount, and error associated with each credit attempted. Each field would end in the number

example:
if total_number_transactions = 3 then in the response there would be a status1 , status2 and a status3 field.

3.3.5.   Retail (Card Present) Single Transaction Credit Response

Returned from XMLGateway for operation_type retail_alone_credit:
```
<?xml version="1.0" encoding="UTF-8"?>
<RESPONSE>
<FIELDS>
<FIELD KEY="status">0-error  1-success  2-declined</FIELD>
<FIELD KEY="auth_response">message from the bank </FIELD>
<FIELD KEY="order_id">echoed back from original post</FIELD>
<FIELD KEY="reference_number">returned for use with credits/voids/settles</FIELD>
<FIELD KEY="error">error text</FIELD>
</FIELDS>
</RESPONSE>
```

3.3.6.   ACH Credit Transaction Response

Returned from XMLGateway for operation_type ach_credit:
```
<?xml version="1.0" encoding="UTF-8"?>
<RESPONSE>
<FIELDS>
<FIELD KEY="status">0-error  1-success  2-declined</FIELD>
<FIELD KEY="auth_response">message from the bank </FIELD>
<FIELD KEY="order_id">echoed back from original post</FIELD>
<FIELD KEY="reference_number">returned for use with credits/voids/settles</FIELD>
<FIELD KEY="error">error text</FIELD>
</FIELDS>
</RESPONSE>
```

3.3.7.   Partial Reversal Transaction

This will attempt to perform a partial credit on the specified transaction entered based on a <reference_number> of a previously authorized transaction without the need to re-input card data or ach check data. This is only applicable for Sale operations that haven't closed and haven't been voided in a retail or moto environment.

Partial Reversal Operations

Field Name	Field Value	Required
Transaction Header Section		
transaction_center_id	Number	X
gateway_id	Unique identifier	X
operation_type	"partial_reverse"	X
mid	String	
tid	String	If mid
processor	String	If mid
processor_id	Numeric	Used in place of mid/tid/processor
Transaction Details Section		
order_id	String	X
total	Money	X
Additional Authorization Data Section		
reference_number	Numeric	X

Reference Number partial_reverse:
```
<?xml version="1.0" encoding="UTF-8"?>
<TRANSACTION>
<FIELDS>
<FIELD KEY="transaction_center_id">1264</FIELD>
<FIELD KEY="gateway_id">a91c38c3-7d7f-4d29-acc7-927b4dca0dbe</FIELD>
<FIELD KEY="operation_type">partial_reverse</FIELD>
<FIELD KEY="order_id">Test Txn Oid</FIELD>
<FIELD KEY="total">1.00</FIELD>
<FIELD KEY="reference_number">125095</FIELD>
</FIELDS>
</TRANSACTION>
```

3.3.8. Partial Reversal Transaction Response

Returned from XMLGateway for operation_type partial_reverse:
```
<?xml version="1.0" encoding="UTF-8"?>
<RESPONSE>
<FIELDS>
<FIELD KEY="status">0-error  1-success  2-declined</FIELD>
<FIELD KEY="auth_response">message from the bank </FIELD>
<FIELD KEY="order_id">echoed back from original post</FIELD>
<FIELD KEY="reference_number">returned for reference</FIELD>
<FIELD KEY="error">error text</FIELD>
</FIELDS>
</RESPONSE>
```

3.4.   *Void Operations*

VOID operations are performed against AUTH transactions that have not settled or ACH transactions that have not posted. Voids will prevent the authorized credit card or pending ACH transactions from ever settling/posting. You cannot VOID a credit card settled or credited transaction nor can a posted ACH transaction be voided.

The <total_number_transactions> field specifies the **N** number of <reference_number> tags to look for. They must match or it will error out.

Field Name	Field Value	Required
Transaction Header Section		
transaction_center_id	Number	X
gateway_id	Unique identifier	X
operation_type	"void" or "ach_void" or "void_partial"	X
mid	String	
tid	String	If mid
processor	String	If mid
processor_id	Numeric	Used in place of mid/tid/processor
Credit/Void/Settle Data Section		
total_number_transactions	String	X
reference_number1…n	numeric	X – if void_partial, then this should be the partial_id retrieved from the original auth/sale.
Customer Ip Address Section		
remote_ip_address	String	

3.4.1.　E-commerce/MOTO/Retail (Card Present)/ACH Void Existing Transaction

```
<?xml version="1.0" encoding="UTF-8"?>
<TRANSACTION>
<FIELDS>
<FIELD KEY="transaction_center_id">1264</FIELD>
<FIELD KEY="gateway_id">a91c38c3-7d7f-4d29-acc7-927b4dca0dbe</FIELD>
<FIELD KEY="operation_type">void</FIELD>
<FIELD KEY="total_number_transactions">2</FIELD>
<FIELD KEY="reference_number1">1222</FIELD>
<FIELD KEY="reference_number2">33334</FIELD>
<FIELD KEY="remote_ip_address">127.0.0.1</FIELD>
</FIELDS>
</TRANSACTION>
```

3.4.2.　E-commerce/MOTO/Retail (Card Present)/ACH Void Existing Transaction Response

Returned from XMLGateway for operation_type void:
```
<?xml version="1.0" encoding="UTF-8"?>
<RESPONSE>
  <FIELDS>
    <FIELD KEY="total_transactions_voided">Total number of transactions voided</FIELD>
```

```
            <FIELD KEY="status1..n"> 0-error  1-success  2-rejected</FIELD>
            <FIELD KEY="response1..n">Response text returned by Transaction Center</FIELD>
            <FIELD KEY="reference_number1..n">reference number of credited transaction</FIELD>
            <FIELD KEY="error1..n">error text</FIELD>
        </FIELDS>
    </RESPONSE>
```

NOTE

total_transactions_voided - the number of successful voids performed on the list of voids attempted. This value can be used to check the successful statuses of the voids attempted.

3.5.   Settle Operations

SETTLE operations allow the merchant to submit the reference numbers of corresponding authorized transactions for batch settlement. The settlement occurs between 2:00 AM and 6:00 AM each day. All AUTH transactions must be followed with a corresponding SETTLE operation against their reference number in order for the merchant to receive the funds for the transaction.

The <total_number_transactions> field specifies the **N** number of <reference_number> and <settle_amount> tags to look for. They must match or it will error out.

Field Name	Field Value	Required
Transaction Header Section		
transaction_center_id	Number	X
gateway_id	Unique identifier	X
operation_type	"credit"	X
mid	String	
tid	String	If mid
processor	String	If mid
processor_id	Numeric	Used in place of mid/tid/processor
Credit/Settle Data Section		
total_number_transactions	String	X
reference_number1…n	numeric	X
settle_amount1…n	Money	X
Customer IP Address Section		
remote_ip_address	String	

3.5.1.   E-commerce/MOTO/Retail (Card Present) Settle Existing Transactions

```
<?xml version="1.0" encoding="UTF-8"?>
<TRANSACTION>
<FIELDS>
<FIELD KEY="transaction_center_id">1264</FIELD>
<FIELD KEY="gateway_id">a91c38c3-7d7f-4d29-acc7-927b4dca0dbe</FIELD>
<FIELD KEY="operation_type">settle</FIELD>
<FIELD KEY="total_number_transactions">2</FIELD>
<FIELD KEY="reference_number1">1222</FIELD>
```

```
<FIELD KEY="reference_number2">33334</FIELD>
<FIELD KEY="settle_amount1">5.00</FIELD>
<FIELD KEY="settle_amount2">1.00</FIELD>
<FIELD KEY="remote_ip_address">127.0.0.1</FIELD>
</FIELDS>
</TRANSACTION>
```

3.5.2.   E-commerce/MOTO/Retail (Card Present) Settle Existing Transaction Response

Returned from XMLGateway for operation_type settle:
```
<?xml version="1.0" encoding="UTF-8"?>
<RESPONSE>
  <FIELDS>
    <FIELD KEY="total_transactions_settled">Total number of transactions settled</FIELD>
    <FIELD KEY="total_amount_settled">amount that was successful set to settle pending</FIELD>
    <FIELD KEY="status1..n"> 0-error  1-success  2-rejected</FIELD>
    <FIELD KEY="response1..n">Response text returned by Transaction Center</FIELD>
    <FIELD KEY="reference_number1..n">reference number of credited transaction</FIELD>
    <FIELD KEY="settle_amount1..n">amount settled w/o $</FIELD>
    <FIELD KEY="batch_number1..n">Batch Number for corresponding transaction</FIELD>
    <FIELD KEY="error1..n">error text</FIELD>
  </FIELDS>
</RESPONSE>
```

NOTE

total_transactions_settled - the number of successful settles performed on the list of settles attempted. This value can be used to check the successful statuses of the returned settles and make sure the number of successful statuses is the same as the total_transactions_settled.
total_amount_settled – the total value of the transactions marked for settle pending which, if all approved will be deposited into your account.

1..n – this means that if the total_number_transactions = 3 which was passed in then there should be a status, response, reference_number, batch_number, settle_amount, and error associated with each settle attempted. Each field would end in the number

ex:
if total_number_transactions = 3 then in the response there would be a status1 , status2 and a status3 field..

3.6.   *Query Operations*

QUERY operations allow the merchant to query the transaction database for all transaction types over a specified query range. The merchant may query any combination of ACH and or Credit Card transaction data. The transaction data is returned with reference numbers. Reference numbers are required to be passed into the gateway in order to process further operations against each transaction. NOTE – the full card number or ACH DDA (direct deposit account) number will not be exposed in a query transaction as per PCI requirements.

Field Name	Field Value	Required
Transaction Header Section		
transaction_center_id	Number	X
gateway_id	Unique identifier	X
operation_type	"query"	X
mid	String	
tid	String	If mid
processor	String	If mid
processor_id	Numeric	Used in place of mid/tid/processor
Query Data Section		
trans_action	String	
card_type	String	
trans_type	Set group	
trans_status	Int	
begin_date	String	X
end_date	String	X
low_amount	Money	
high_amount	Money	
order_id	String	
card_number	numeric	

```xml
<?xml version="1.0" encoding="UTF-8"?>
<TRANSACTION>
  <FIELDS>
    <FIELD KEY="transaction_center_id">1264</FIELD>
    <FIELD KEY="gateway_id">a91c38c3-7d7f-4d29-acc7-927b4dca0dbe</FIELD>
    <FIELD KEY="operation_type">query</FIELD>
    <FIELD KEY="trans_action">CC</FIELD>
    <FIELD KEY="card_type"> </FIELD>
    <FIELD KEY="trans_type">SALE</FIELD>
    <FIELD KEY="trans_status">1</FIELD>
    <FIELD KEY="begin_date">120108</FIELD>
    <FIELD KEY="begin_time"> </FIELD>
    <FIELD KEY="end_date"> </FIELD>
    <FIELD KEY="end_time"> </FIELD>
    <FIELD KEY="order_id">test txn</FIELD>
    <FIELD KEY="card_number"></FIELD>
    <FIELD KEY="low_amount">1.00</FIELD>
    <FIELD KEY="high_amount">500.00</FIELD>
  </FIELDS>
</TRANSACTION>
```

NOTES
for any of the not required fields if you leave them blank they query will search for all possible options and ignore the blank fields.

trans_status: 2 implies successful and will only return transactions that can be submitted for settlement via the settle operation.

Returned from XMLGateway for operation_type query:

```
<?xml version="1.0" encoding="UTF-8"?>
<RESPONSE>
  <FIELDS>
     <FIELD KEY="records_found">total number of records returned </FIELD>
     <FIELD KEY="status"> 0-error  1-success  2-rejected</FIELD>
     <FIELD KEY="trans_type1..n">type of trans </FIELD>
     <FIELD KEY="trans_status1..n">status</FIELD>
     <FIELD KEY="settled1..n">1 (yes) or 0 (no)</FIELD>
     <FIELD KEY="credit_void1..n"> None, Full Credit, Partial Credit, Void </FIELD>
     <FIELD KEY="order_id1..n">unique id </FIELD>
     <FIELD KEY="reference_number1..n">unique id </FIELD>
     <FIELD KEY="trans_time1..n">MM/DD/YYYY HH:MM:SS AM/PM</FIELD>
     <FIELD KEY="card_type1..n">( Visa, Amex, Discover or MasterCard) </FIELD>
     <FIELD KEY="amount1..n">auth or sale amount with no $</FIELD>
     <FIELD KEY="amount_settled1..n">settled amount with no $</FIELD>
     <FIELD KEY="amount_credited1..n">credited amount with no $</FIELD>
     <FIELD KEY="posted_by1..n">username that performed the transaction</FIELD>
     <FIELD KEY="signature_url1..n">URL for the electronic signature image</FIELD>
     <FIELD KEY="error1..n">error text</FIELD>
     </FIELDS>
</RESPONSE>
```

NOTE
if records_found is 0 then only error is returned no error1...n

trans_status value in response will be 0 or 1 only for success or failure. If the initial query filtered for trans_status = 2, all the response records will have a trans_status = 1.

3.7. *CIM Operations*

CIM operations are separate functions from the CIM authorization and sale warranting its own section in the documentation. The operations available under the API for CIM are CIM_INSERT, CIM_EDIT and CIM_DELETE.

CIM_INSERT allows the creation of a CIM account without first charging a customer. A CIM account can be created with Credit Card information and/or ACH information and Shipping information.

CIM_EDIT allows the changing of any or all fields associated to a CIM records based on the CIIM reference number. An example requiring a CIM edit would be updating a credit card expiration date. Instead of issuing a brand new CIM Sale, a CIM_EDIT can be performed by sending the new credit card expiration date in order to update the record and continue to be able to perform auth/sale operations.

CIM_DELETE allows the permanent removal of a CIM record from the system. By sending a CIM_DELETE command to the gateway, the record associated to the CIM reference number and all of its associated records will be removed from storage. NOTE – CIM_DELETE cannot be undone. Once the command mis sent and a successful response is received, the record is removed permanently without the ability to restore.

3.7.1. CIM INSERT Creating Customer

CIM Field Requirements Create Customer

Field Name	Field Value	Required
Transaction Header Section		
transaction_center_id	Number	X
gateway_id	Unique identifier	X
Operation_type	"cim_insert"	X
CIM Details Section		
cim_ref_num	String	X
Shipping Address Section		
shipping_name	String	
shipping_street	String	
shipping_street2	String	
shipping_city	String	
shipping_state	String	
shipping_zip	String	
shipping_country	String	
shipping_phone	String	
shipping_email	String	
shipping_method	String	
Credit Card Data Section		
card_name	String	
card_number	Numeric	
card_exp	Numeric	
ACH Data Section		
aba	Numeric	
dda	Numeric	
ach_account_type	Char	
ach_name	String	
Card Holder/ACH Owner Billing Address Section		
owner_name	String	x
owner_street	String	x
owner_street2	String	
owner_city	String	x
owner_state	String	x
owner_zip	String	x
owner_country	String	x
owner_email	String	
owner_phone	String	

Example: CIM INSERT

```
<?xml version="1.0" encoding="UTF-8"?>
<TRANSACTION>
<FIELDS>
<FIELD KEY="transaction_center_id">1264</FIELD>
<FIELD KEY="gateway_id">a91c38c3-7d7f-4d29-acc7-927b4dca0dbe</FIELD>
<FIELD KEY="operation_type">cim_insert </FIELD>
<FIELD KEY="cim_ref_num">100231</FIELD>
<FIELD KEY="shipping_name">Bob Tester</FIELD>
<FIELD KEY="shipping_street">123 Test Rd.</FIELD>
```

```
<FIELD KEY="shipping_street2"></FIELD>
<FIELD KEY="shipping_city">Cityville</FIELD>
<FIELD KEY="shipping_state">NJ</FIELD>
<FIELD KEY="shipping_zip">08035</FIELD>
<FIELD KEY="shipping_country">US</FIELD>
<FIELD KEY="shipping_email"></FIELD>
<FIELD KEY="shipping_phone">555-555-5555</FIELD>
<FIELD KEY="shipping_method">UPS</FIELD>
<FIELD KEY="card_name">Visa</FIELD>
<FIELD KEY="card_number">4111111111111111</FIELD>
<FIELD KEY="card_exp">1018</FIELD>
<FIELD KEY="owner_name">Bob Tester</FIELD>
<FIELD KEY="owner_street">123 Test Rd.</FIELD>
<FIELD KEY="owner_street2"></FIELD>
<FIELD KEY="owner_city">Cityville</FIELD>
<FIELD KEY="owner_state">NJ</FIELD>
<FIELD KEY="owner_zip">08035</FIELD>
<FIELD KEY="owner_country">US</FIELD>
<FIELD KEY="owner_email"></FIELD>
<FIELD KEY="owner_phone">555-555-5555</FIELD>
<FIELD KEY="aba">031200213</FIELD>
<FIELD KEY="dda">7596321546</FIELD>
<FIELD KEY="ach_account_type">C</FIELD>
<FIELD KEY="ach_name">Citibank </FIELD>
</FIELDS>
</TRANSACTION>
```

3.7.2. CIM INSERT Transaction Response

Returned from XMLGateway for operation_type cim_insert:
```
<?xml version="1.0" encoding="UTF-8"?>
<RESPONSE>
  <FIELDS>
    <FIELD KEY="status"> 0-error  1-success  2-Rejected</FIELD>
    <FIELD KEY="cim_ref_num">cim reference number</FIELD>
    <FIELD KEY="error">error text</FIELD>
  </FIELDS>
</RESPONSE>
```

3.7.3. CIM EDIT Existing Customer

CIM Field Requirements Edit Existing Customer

Field Name	Field Value	Required
Transaction Header Section		
transaction_center_id	Number	X
gateway_id	Unique identifier	X
Operation_type	"cim_edit"	X
CIM Details Section		
cim_ref_num	String	X
Shipping Address Section		

shipping_name	String	
shipping_street	String	
shipping_street2	String	
shipping_city	String	
shipping_state	String	
shipping_zip	String	
shipping_country	String	
shipping_phone	String	
shipping_email	String	
shipping_method	String	
Credit Card Data Section		
card_name	String	
card_number	Numeric	
mag_data	String	
card_exp	Numeric	
cvv2	Numeric	
card_sequence	Int	
ACH Data Section		
aba	Numeric	
dda	Numeric	
ach_account_type	Char	
ach_sequence	Int	
Card Holder/ACH Owner Billing Address Section		
owner_name	String	
owner_street	String	
owner_street2	String	
owner_city	String	
owner_state	String	
owner_zip	String	
owner_country	String	
owner_email	String	
owner_phone	String	

Example: CIM EDIT (Card Expiration and Card Holder Address)

```
<?xml version="1.0" encoding="UTF-8"?>
<TRANSACTION>
<FIELDS>
<FIELD KEY="transaction_center_id">1264</FIELD>
<FIELD KEY="gateway_id">a91c38c3-7d7f-4d29-acc7-927b4dca0dbe</FIELD>
<FIELD KEY="operation_type">cim_edit </FIELD>
<FIELD KEY="card_exp">1018</FIELD>
<FIELD KEY="owner_street">789 New CIM Rd.</FIELD>
<FIELD KEY="owner_city">New CIM City</FIELD>
<FIELD KEY="cim_ref_num">100231</FIELD>
</FIELDS>
</TRANSACTION>
```

3.7.4. CIM DELETE Existing Customer

CIM Field Requirements Edit Existing Customer

Field Name	Field Value	Required
Transaction Header Section		
transaction_center_id	Number	X
gateway_id	Unique identifier	X
Operation_type	"cim_delete"	X
CIM Details Section		
cim_ref_num	String	X

Example: CIM DELETE Existing Customer
<?xml version="1.0" encoding="UTF-8"?>
<TRANSACTION>
<FIELDS>
</FIELDS>
</TRANSACTION>

3.7.5. CIM EDIT/DELETE Transaction Response

Returned from XMLGateway for operation_type cim_edit/cim_delete:
<?xml version="1.0" encoding="UTF-8"?>
<RESPONSE>
  <FIELDS>
  </FIELDS>
</RESPONSE>

3.7.6. CIM QUERY Existing Customer

CIM Field Requirements Query Existing Customer

Field Name	Field Value	Required
Transaction Header Section		
transaction_center_id	Number	X
gateway_id	Unique identifier	X
Operation_type	"cim_query"	X
CIM Details Section (at least 1 is required)		
cim_ref_num	String	
customer_name	String	
customer_email	String	

Example: CIM QUERY Existing Customer
<?xml version="1.0" encoding="UTF-8"?>
<TRANSACTION>
<FIELDS>

```
<FIELD KEY="operation_type">cim_query</FIELD>
<FIELD KEY="cim_ref_num">1009</FIELD>
<FIELD KEY="customer_name"></FIELD>
<FIELD KEY="customer_email"></FIELD>
</FIELDS>
</TRANSACTION>
```

3.7.7. CIM QUERY Existing Customer Response

This response is slightly different then other responses. It contains a wrapper block <cim_record> which can contain 0 or more of the following blocks: <card_sequences>, >ach_sequences> and <ship_sequences>. Each of these blocks can contain 1 or more sequence records. In the return sample below, <card_sequences> and <ship_sequences> each contain two records whereas the <ach_sequences> block only contains one.

The sequence numbers for each record can be used to perform CIM EDIT operations and other CIM operations where you would like to specify the default sequence record instead of defaulting to sequence 0.

NOTE: Sensitive data such as the card number and ACH account number is limited to the last 4 characters.

Returned from XMLGateway for operation_type cim_query:
```
<?xml version="1.0" encoding="UTF-8"?>
<RESPONSE>
  <FIELDS>
     <FIELD KEY="status"> 0-error  1-success  2-Rejected</FIELD>
     <FIELD KEY="error">error text</FIELD>
<cim_record>
   <card_work_phone />
   <card_street>123 st</card_street>
   <card_cell_phone />
   <card_street2 />
   <date_created>6/8/2009 1:57:19 PM</date_created>
   <cim_ref_num>1009</cim_ref_num>
   <storename>demo</storename>
   <card_country>usa</card_country>
   <card_city>My city</card_city>
   <name_on_card>Test Name</name_on_card>
   <card_phone>555-555-5555</card_phone>
   <card_zip>08080</card_zip>
   <card_email>test@test.com</card_email>
   <card_state>NJ</card_state>
   <card_sequences>
    <card_sequence>
     <cim_ref_num>1009</cim_ref_num>
     <card_num>1110</card_num>
     <card_exp>0119</card_exp>
     <card_type>visa</card_type>
     <card_sequence>0</card_sequence>
    </card_sequence>
```

```xml
    <card_sequence>
     <cim_ref_num>1009</cim_ref_num>
     <card_num>1111</card_num>
     <card_exp>1010</card_exp>
     <card_type>Visa</card_type>
     <card_sequence>1</card_sequence>
    </card_sequence>
   </card_sequences>
   <ach_sequences>
    <ach_sequence>
     <cim_ref_num>1009</cim_ref_num>
     <ach_account_type>C</ach_account_type>
     <aba>0053</aba>
     <dda>5456</dda>
     <name>P</name>
     <ach_sequence>0</ach_sequence>
    </ach_sequence>
   </ach_sequences>
   <shipping_sequences>
    <shipping_sequence>
     <cim_ref_num>1009</cim_ref_num>
     <ship_name>bob thomas</ship_name>
     <ship_street>123 test rd</ship_street>
     <ship_street2 />
     <ship_city />
     <ship_state />
     <ship_zip>92835</ship_zip>
     <ship_country />
     <ship_phone />
     <ship_email />
     <ship_method />
     <ship_sequence>0</ship_sequence>
    </shipping_sequence>
    <shipping_sequence>
     <cim_ref_num>1009</cim_ref_num>
     <ship_name>Test Name</ship_name>
     <ship_street>123 st</ship_street>
     <ship_street2 />
     <ship_city>My city</ship_city>
     <ship_state>NJ</ship_state>
     <ship_zip>08080</ship_zip>
     <ship_country>usa</ship_country>
     <ship_phone>555-555-5555</ship_phone>
     <ship_email>test@test.com</ship_email>
     <ship_method />
     <ship_sequence>1</ship_sequence>
    </shipping_sequence>
   </shipping_sequences>
  </cim_record>
  </FIELDS>
 </RESPONSE>
```

3.8.   *Recurring Billing Operations*

The RECURRING BILLING operation is used to create new recurring billing transactions from an existing authorization, or to enable/disable/update existing recurring billing transactions.

3.8.1.   Create/Modify Recurring Billing Transaction

Field Name	Field Value	Required
Transaction Header Section		
transaction_center_id	Number	X
gateway_id	Unique identifier	X
operation_type	"recurring_modify"	X
processor	String	X
mid	String	
tid	String	If mid
processor_id	Numeric	Used in place of mid/tid/processor
Details Section		
reference_number	Numeric	X
is_ach	1 or 0	X
recurring	1 or 0	
recurring_type	Daily, weekly, biweekly, monthly, bimonthly, quarterly, semiannually, annually	Required if setting up an existing authorization as recurring for the first time.
recurring_start_date	MM/DD/YYYY	
recurring_end_date	MM/DD/YYYY	
card_exp	MMYY	Only used if processing a credit card transaction.
total	Numeric	
owner_zip	String	
owner_email	String	
owner_phone	String	
send_invoice	1 or 0	
aba	Numeric	Only used if processing an ACH transaction.
dda	Numeric	Only used if processing an ACH transaction.

Example:
```
<TRANSACTION>
<FIELDS>
<FIELD KEY="transaction_center_id">1264</FIELD>
<FIELD KEY="gateway_id"> a91c38c3-7d7f-4d29-acc7-927b4dca0dbe </FIELD>
<FIELD KEY="operation_type">recurring_modify</FIELD>
<FIELD KEY="reference_number">19411751</FIELD>
<FIELD KEY="recurring_start_date">05/01/2011</FIELD>
<FIELD KEY="recurring_end_date">07/05/2012</FIELD>
<FIELD KEY="total">5.00</FIELD>
<FIELD KEY="recurring_type">monthly</FIELD>
<FIELD KEY="is_ach">0</FIELD>
</FIELDS>
</TRANSACTION>
```

3.8.2. RECURRING_MODIFY Response

Response for Recurring_modify
```
<?xml version="1.0" encoding="utf-8"?>
<RESPONSE>
 <FIELDS>
   <FIELD KEY="status">0-error 1-success</FIELD>
   <FIELD KEY="error">error text</FIELD>
 </FIELDS>
</RESPONSE>
```

4. Misc. Operations

4.1.1. Check if Card Number is a Debit Card

CIM Field Requirements Edit Existing Customer

Field Name	Field Value	Required
Transaction Header Section		
transaction_center_id	Number	X
gateway_id	Unique identifier	X
Operation_type	"is_debit_card"	X
Details Section		
Card_number	String	X

Example: is_debit_card check
```
<?xml version="1.0" encoding="UTF-8"?>
<TRANSACTION>
<FIELDS>
<FIELD KEY="transaction_center_id">1264</FIELD>
<FIELD KEY="gateway_id"> a91c38c3-7d7f-4d29-acc7-927b4dca0dbe</FIELD>
<FIELD KEY="operation_type">is_debit_card</FIELD>
<FIELD KEY="cim_ref_num">4111111111111111</FIELD>
</FIELDS>
</TRANSACTION>
```

Returned from XMLGateway for operation_type is_debit_card:
```
<?xml version="1.0" encoding="UTF-8"?>
<RESPONSE>
  <FIELDS>
    <FIELD KEY="status"> 0-error  1-yes</FIELD>
    <FIELD KEY="error">error text</FIELD>
  </FIELDS>
</RESPONSE>
```

5. XML Error Responses

Malformed fields or bad xml in all cases will be returned in the xml response defined for the operation type sent in with the error field defined. For operation types that cannot be induced from incoming fields a generic xml response will come back with simply an error field defined with the reason for the failure.

As is commonplace in XML, fields with no values can be self closing, if no error is returned in the error field, then it will return as <FIELD KEY="error" />. Please note that some successful transactions will return the error field empty with the self closing syntax.

STATUS – 0:

Notice most fields are blank when status = 0! If the status is a 1 or 2 then the fields will be filled in with the appropriate responses

error –
As is commonplace in XML, fields with no values can be self closing, if no error is returned in the error field, then it will return as <FIELD KEY="error" />. Please note that some successful transactions will return the error field empty with the self closing syntax.

If status is 1 the response will look like this:
<FIELD KEY="error" />

reference_number – this is the number used in later operations to reference specific transactions for credits, voids and settles.

Generic Example Error:

<?xml version="1.0" encoding="UTF-8"?>
<RESPONSE>
  <FIELDS>
    <FIELD KEY="error">DESCRIPTIVE ERROR MESSAGE</FIELD>
  </FIELDS>
</RESPONSE>

6. Test Authorization Account Information

6.1.1.   Testing of XML Gateway (Not Approved GoEmerchant Customers)

If you are implementing the gateway prior to becoming a GoEmerchant customer, you may test your code for the XML Gateway by supplying test transaction data that will either produce an authorization, decline or error. By using the test data, you may ensure that you are properly communicating with the transaction gateway and also properly parsing the return values.

To test a successful response from the gateway, please provide the values for the following variables:

merchant = 1264
password = password
gateway_id = a91c38c3-7d7f-4d29-acc7-927b4dca0dbe
        order_id = any unique invoice number.

To get an approved response for Credit Card, use the following information:

Visa: 4111111111111111	CVV2 = 123
Mastercard: 5000300020003003	CVC2 = 123
Discover: 6011111111111117	CVV2 = 123
Amex: 374255312721002	CVV2 = 1234

To get a pending response for ACH, use the following information:
aba (Routing Number): 123123123
dda (Direct Deposit Account): 1234567890
ach_name: Any Bank
ach_category_text : Web Payments
ach_account_type: C
close_date: NOTE –please be sure to use a post date that is either the current date or in the future.

Test Address information:
Address: 123 Test St
Zip: 12345-6789

The remaining field values you can make whatever you want to test all the various aspects of the API, just make sure to fulfill the defined criteria listed above for the different operation types.

* As this is a generic test account, try to use a formula that will generate unique order_id's based on your merchant name.

To test a decline Credit Card response from the gateway, please provide the values for the following variables:

Use above information and any other card number other than those listed above. Legitimate card numbers will decline since this is a test account.

The remaining field values you can make whatever you want to test all the various aspects of the API, just make sure to fulfill the defined criteria listed above for the different operation types.

To test an error response from the gateway, please provide the values for the following variables:

To generate an error, use the above account information and provide bad XML or bad data in the XML fields and an error response will be generated.

The remaining field values you can make whatever you want to test all the various aspects of the API, just make sure to fulfill the defined criteria listed above for the different operation types.

NOTE – It is strongly recommended that all procedures be tested to ensure that your gateway integration is complete and correct.

6.1.2. Sandbox Testing of XML Gateway (Approved GoEmerchant Customers)

For those customers that have been boarded by GoEmerchant and already have received their Transaction Center Login information as well as approved ISO/ISV integrators, a complete "Sandbox" test environment exists providing the end user with an interface to completely test all transaction responses from authorization to batch. The test transactions are completed using a set of test card numbers and predefined authorization amounts. The predefined amounts determine the response returned from the gateway.

In order to use the Sandbox environment, you must download a copy of the documentation from the Virtual Terminal located under the Support/Downloads menu.

7. Sample Code

The sample code is provided as is. The sample provides an XML interface to each of the operation_types in one process. This code can be cut out and saved on any windows platform with perl 5 installed and the XML::Simple and Win32::OLE modules installed. Only order_id and reference fields need necessarily be changed. The sample uses the test gateway account.

7.1.1. C#.NET Sample for E-commerce Auth

```
using System;
using System.Data;
using System.Configuration;
using System.Collections;
using System.Web;
using System.Web.Security;
using System.Web.UI;
using System.Web.UI.WebControls;
using System.Web.UI.WebControls.WebParts;
using System.Web.UI.HtmlControls;

//need these includes to perform the request
using System.Net;
using System.IO;
using System.Text;
using System.Xml;
using System.Collections;

namespace gateway_example
{
  public partial class _Default : System.Web.UI.Page
  {
    protected void Page_Load(object sender, EventArgs e)
    {
      Random rgen = new Random();

      Response.Write("This will send a get request to the xmlgateway and attempt to perform an auth operation<br><hr><br>");

      string lcUrl = "https://secure.goemerchant.com/secure/gateway/xmlgateway.aspx";

      HttpWebRequest loHttp = (HttpWebRequest)WebRequest.Create(lcUrl);
```

```csharp
// *** Send any POST data
string lcPostData =
  @"<?xml version=""1.0"" encoding=""UTF-8""?>
  <TRANSACTION>
   <FIELDS>
    <FIELD KEY=""transaction_center_id"">1264</FIELD>
    <FIELD KEY=""gateway_id"">a91c38c3-7d7f-4d29-acc7-927b4dca0dbe</FIELD>
    <FIELD KEY=""operation_type"">auth</FIELD>
    <FIELD KEY=""order_id"">ectest " + rgen.Next(0, 1000) + rgen.Next(1000, 5000) + @"</FIELD>
    <FIELD KEY=""total"">1.00</FIELD>
    <FIELD KEY=""card_name"">Visa</FIELD>
    <FIELD KEY=""card_number"">4111111111111111</FIELD>
    <FIELD KEY=""card_exp"">1020</FIELD>
    <FIELD KEY=""cvv2""></FIELD>
    <FIELD KEY=""owner_name"">Bob Tester</FIELD>
    <FIELD KEY=""owner_street"">123 Test Rd</FIELD>
    <FIELD KEY=""owner_city"">Cityville</FIELD>
    <FIELD KEY=""owner_state"">PA</FIELD>
    <FIELD KEY=""owner_zip"">19036</FIELD>
    <FIELD KEY=""owner_country"">US</FIELD>
    <FIELD KEY=""owner_email""></FIELD>
    <FIELD KEY=""owner_phone""></FIELD>
    <FIELD KEY=""recurring"">0</FIELD>
    <FIELD KEY=""recurring_type""></FIELD>
    <FIELD KEY=""remote_ip_address"">" + Request.ServerVariables["REMOTE_ADDR"].ToString() + @"</FIELD>
   </FIELDS>
  </TRANSACTION>";

//done for visible printout
Response.Write("<b>Message Being Sent:</b><br><hr><br>");
Response.Write("<pre>" + lcPostData.Replace("<", "&lt;").Replace(">", "&gt;") + "</pre>");

loHttp.Method = "POST";
byte[] lbPostBuffer = System.Text.Encoding.GetEncoding(1252).GetBytes(lcPostData);
loHttp.ContentLength = lbPostBuffer.Length;

Stream loPostData = loHttp.GetRequestStream();
loPostData.Write(lbPostBuffer, 0, lbPostBuffer.Length);
loPostData.Close();

HttpWebResponse loWebResponse = (HttpWebResponse)loHttp.GetResponse();

Encoding enc = System.Text.Encoding.GetEncoding(1252);

StreamReader loResponseStream = new StreamReader(loWebResponse.GetResponseStream(), enc);

string lcHtml = loResponseStream.ReadToEnd();

loWebResponse.Close();
loResponseStream.Close();

//done for visible printout
Response.Write("<br><hr><br><b>Raw Response:</b><br><hr><br>");
Response.Write("<pre>" + lcHtml.Replace("<FIELD ", "\n<FIELD ").Replace("<", "&lt;").Replace(">", "&gt;") + "</pre>");


Response.Write("<br><hr><br><b>Parsed XML Response:</b><br><hr><br>");
Hashtable xml_hash = parseXML(lcHtml);
if (xml_hash != null && xml_hash.Count > 0)
{
  foreach (string k in xml_hash.Keys)
  {
    Response.Write(k + " <=> " + xml_hash[k].ToString() + "<br>");
  }
}
else
```

```
        {
            Response.Write("No XML was parsed");
        }
    }
}

/// <summary>
/// takes in raw xml string and attempts to parse it into a workable hash.
/// all valid xml for the gateway contains
/// <transaction><fields><field key="attribute name">value</field></fields></transaction>
/// there will be 1 or more (should always be more than 1 to be valid) field tags
/// this method will take the attribute name and make that the hash key and then the value is the value
/// if an error occurs then the error key will be added to the hash.
/// </summary>
/// <param name="xml"></param>
/// <returns></returns>
private Hashtable parseXML(string xml)
{
    Hashtable ret_hash = new Hashtable(); //stores key values to return
    XmlTextReader txtreader = null;
    XmlValidatingReader reader = null;

    if (xml != null && xml.Length > 0)
    {

        try
        {
            //Implement the readers.
            txtreader = new XmlTextReader(new System.IO.StringReader(xml));
            reader = new XmlValidatingReader(txtreader);

            //Parse the XML and display the text content of each of the elements.
            while (reader.Read())
            {
                if (reader.IsStartElement() && reader.Name.ToLower() == "field")
                {
                    if (reader.HasAttributes)
                    {
                        //we want the key attribute value
                        ret_hash[reader.GetAttribute(0).ToLower()] = reader.ReadString();
                    }
                    else
                    {
                        ret_hash["error"] = "All FIELD tags must contains a KEY attribute.";
                    }
                }
            } //ends while
        }
        catch (Exception e)
        {
            //handle exceptions
            ret_hash["error"] = e.Message;
        }
        finally
        {
            if (reader != null)
                reader.Close();
        }
    }
    else
    {
        //incoming xml is empty
        ret_hash["error"] = "No data was present. Valid XML must be sent in order to process a transaction.";
    }

    return ret_hash;
} //ends parseXML
```

```
        }
}
```

7.1.2.   Perl 5 Sample for E-commerce

```
Windows Perl 5.6 – Object Method
######     XMLgatewaytest.cgi  ########################################
#
#This program is used to connect to the xml gateway api
#you will notice 6 variables named $xml_1 - $xml_6
#each variable contains the correctly formatted xml request required to send to the api
#for each of the 6 operation types that can be performed.
#Each request can be altered by you to test the api and validate the responses you receive
#back.
#
#You will also notice the two code snippets that handle seding the XML to the api
#and receiving the response back. Simply change which XML variable you want to
#either POST or GET to test the different operation types and responses
#########  Explanation Over  ########################################
#!c:\perl5\bin\perl.exe

use strict;
use warnings;

$|=1;

# Perl library modules
use WIN32::OLE;
use XML::Simple;

print "Content-type:text/html\n\n";

my $xml_1 = '
<?xml version="1.0" encoding="UTF-8"?>
<TRANSACTION>
        <FIELDS>
                <FIELD KEY="transaction_center_id">1264</FIELD>
                <FIELD KEY="gateway_id">a91c38c3-7d7f-4d29-acc7-927b4dca0dbe</FIELD>
                <FIELD KEY="operation_type">sale</FIELD>
                <FIELD KEY="order_id">YOURID_NUMBER</FIELD>
                <FIELD KEY="total">5.00</FIELD>
                <FIELD KEY="card_name">Visa</FIELD>
                <FIELD KEY="card_number">4111111111111111</FIELD>
                <FIELD KEY="card_exp">1106</FIELD>
                <FIELD KEY="cvv2">123</FIELD>
                <FIELD KEY="owner_name">Bob Auth</FIELD>
                <FIELD KEY="owner_street">123 Test St</FIELD>
                <FIELD KEY="owner_city">city</FIELD>
                <FIELD KEY="owner_state">PA</FIELD>
                <FIELD KEY="owner_zip">12345-6789</FIELD>
                <FIELD KEY="owner_country">US</FIELD>
                <FIELD KEY="recurring">0</FIELD>
                <FIELD KEY="recurring_type">annually</FIELD>
        </FIELDS>
</TRANSACTION>';

my $xml_2 = '<?xml version="1.0" encoding="UTF-8"?>
<TRANSACTION>
        <FIELDS>
                <FIELD KEY="transaction_center_id">1264</FIELD>
                <FIELD KEY="gateway_id">a91c38c3-7d7f-4d29-acc7-927b4dca0dbe</FIELD>
                <FIELD KEY="operation_type">credit</FIELD>
```

```
                    <FIELD KEY="total_number_transactions">1</FIELD>
                    <FIELD KEY="reference_number1">REF ID FROM SALE </FIELD>
                    <FIELD KEY="credit_amount1">1.00</FIELD>
          </FIELDS>
</TRANSACTION>';


my $xml_3 = '<?xml version="1.0" encoding="UTF-8"?>
<TRANSACTION>
          <FIELDS>
                    <FIELD KEY="transaction_center_id">1264</FIELD>
                    <FIELD KEY="gateway_id">a91c38c3-7d7f-4d29-acc7-927b4dca0dbe</FIELD>
                    <FIELD KEY="operation_type">void</FIELD>
                    <FIELD KEY="total_number_transactions">3</FIELD>
                    <FIELD KEY="reference_number1">REF ID FROM AUTH</FIELD>
                    <FIELD KEY="reference_number2">REF ID FROM AUTH</FIELD>
                    <FIELD KEY="reference_number3">REF ID FROM AUTH</FIELD>
</FIELDS>
</TRANSACTION>';


my $xml_4 = '<?xml version="1.0" encoding="UTF-8"?>
<TRANSACTION>
          <FIELDS>
                    <FIELD KEY="transaction_center_id">1264</FIELD>
                    <FIELD KEY="gateway_id">a91c38c3-7d7f-4d29-acc7-927b4dca0dbe</FIELD>
                    <FIELD KEY="operation_type">settle</FIELD>
                    <FIELD KEY="total_number_transactions">2</FIELD>
                    <FIELD KEY="reference_number1">REF ID FROM AUTH</FIELD>
                    <FIELD KEY="settle_amount1">5.00</FIELD>
                    <FIELD KEY="reference_number2">REF ID FROM AUTH</FIELD>
                    <FIELD KEY="settle_amount2">2.00</FIELD>
          </FIELDS>
</TRANSACTION>';


my $xml_5 = '
<?xml version="1.0" encoding="UTF-8"?>
<TRANSACTION>
          <FIELDS>
                    <FIELD KEY="transaction_center_id">1264</FIELD>
                    <FIELD KEY="gateway_id">a91c38c3-7d7f-4d29-acc7-927b4dca0dbe</FIELD>
                    <FIELD KEY="operation_type">auth</FIELD>
                    <FIELD KEY="order_id">YOURID_NUMBER</FIELD>
                    <FIELD KEY="total">5.00</FIELD>
                    <FIELD KEY="card_name">Visa</FIELD>
                    <FIELD KEY="card_number">4111111111111111</FIELD>
                    <FIELD KEY="card_exp">1006</FIELD>
                    <FIELD KEY="cvv2">123</FIELD>
                    <FIELD KEY="owner_name">Bob Recurring_Sale</FIELD>
                    <FIELD KEY="owner_street">123 test st</FIELD>
                    <FIELD KEY="owner_city">city</FIELD>
                    <FIELD KEY="owner_state">PA</FIELD>
                    <FIELD KEY="owner_zip">12345-6789</FIELD>
                    <FIELD KEY="owner_country">US</FIELD>
                    <FIELD KEY="recurring">0</FIELD>
                    <FIELD KEY="recurring_type"></FIELD>
          </FIELDS>
</TRANSACTION>';

my $xml_6 = '<?xml version="1.0" encoding="UTF-8"?>
<TRANSACTION>
          <FIELDS>
                    <FIELD KEY="transaction_center_id">1264</FIELD>
                    <FIELD KEY="gateway_id">a91c38c3-7d7f-4d29-acc7-927b4dca0dbe</FIELD>
                    <FIELD KEY="operation_type">query</FIELD>
```

```
                        <FIELD KEY="card_type"></FIELD>
                        <FIELD KEY="trans_type">SALE</FIELD>
                        <FIELD KEY="trans_status">0</FIELD>
                        <FIELD KEY="begin_date">100103</FIELD>
                        <FIELD KEY="begin_time">1222AM</FIELD>
                        <FIELD KEY="end_date">123103</FIELD>
                        <FIELD KEY="end_time">1159PM</FIELD>
                        <FIELD KEY="order_id"></FIELD>
                        <FIELD KEY="card_number"></FIELD>
                        <FIELD KEY="low_amount"></FIELD>
                        <FIELD KEY="high_amount"></FIELD>
                </FIELDS>
        </TRANSACTION>';


        my $SendObject = Win32::OLE->new('microsoft.XMLhttp');

        $SendObject->open("POST", "https://secure.goemerchant.com/secure/gateway/xmlgateway.aspx", "false");
        $SendObject->setRequestHeader("Content-type", "text/xml");
        $SendObject->send();
        my $response = $SendObject->responseText;

        print "<html><head><title>testing xml gateway</title><head><body>";
        print "GET RESPONSE: $response";

        #contains key value pairs of xml returned
        my %xml_pairs_get = &ParseXml($response);

        $SendObject->open("POST", "https://secure.goemerchant.com/secure/gateway/xmlgateway.aspx ", "false");
        $SendObject->setRequestHeader("Content-type", "text/xml");

        $SendObject->send($xml_2);
        $response = $SendObject->responseText;

        print "<br><br>POST RESPONSE: $response";

        #contains key value pairs of xml returned
        my %xml_pairs_post = &ParseXml($response);

        print "</body></html>";

        #this method will parse the xml and return a hash of key value pairs
        #which can be manipulated in any way you need them to be
        sub ParseXml {
                my $xml = $_[0];
                my %RESULT;
                my $ref = eval { XMLin($xml) };
                if ($@) {
                        $RESULT{error} = $@;
                        return(%RESULT);
                }


                foreach my $key(@{$ref->{'FIELDS'}{'FIELD'}}) {
                        my ($val1, $val2) = each %$key;
                        my ($val3, $val4) = each %$key;
                        my ($val5, $val6) = each %$key;
                        if($val5 && $val6){
                                $RESULT{$val6} = $val4;
                        }
                        else{
                                $RESULT{$val4} = $val2;
                        }
                }
                return(%RESULT);
        }
```

#EOF