worldpay | integrated payments

Worldpay Integrated Payments

Payment Account Secure Storage (PASS) Overview

Version 1.0.3

July 2019

Product Overview

Worldpay Express™ Interface Specification
Disclaimer, Terms of Use and Compliance Representations

Disclaimer

This **Interface Specification** and all documentation contained herein or provided to you hereunder (the "Specifications") are licensed by Worldpay Integrated Payments, LLC ("WIP"), ("Licensor") on an "AS IS" basis. No representations or warranties are expressed or implied, including, but not limited to, warranties of suitability, quality, merchantability, or fitness for a particular purpose (irrespective of any course of dealing, custom or usage of trade), and all such warranties are expressly and specifically disclaimed. Licensor shall have no liability or responsibility to you or any other person or entity with respect to any liability, loss, or damage, including lost profits whether foreseeable or not, or other obligation for any cause whatsoever, caused or alleged to be caused directly or indirectly by the Specifications. Use of the Specifications signifies agreement with the disclaimer set forth in this paragraph and the below license and restricted use terms and conditions.

Ownership and Restricted Terms of Use

Ownership of all Specifications, related documentation and all intellectual property rights therein and thereto shall remain at all times with Licensor. All rights not expressly granted to you herein are reserved to Licensor. Licensor grant you the right to use the Specification for the sole purpose of transmitting transactions directly to WIP from a merchant transaction originating device. Under no circumstances may you reverse engineer, translate, re-direct, emulate, disseminate to other entities, decompile, adapt, or disassemble the information contained in the Specifications nor shall you attempt to create source code/object code to emulate the Express platform. You agree that the Specifications and the printed materials and documentation that accompany these Specifications are the confidential information of Licensor and may not be used except as otherwise expressly permitted herein.

Compliance Representations

You represent, warrant and agree:

- To comply with the card brand operating regulations and all applicable PCI Data Security Standards ("PCI DSS") and Payment Application Data Security Standards ("PA-DSS") and all requirements applicable to the distribution and use of these Specifications, Worldpay IP products, and your payment application.
- To comply with all applicable federal, state and local laws, rules and regulations, including those related or pertaining to privacy and truncating and masking cardholder account information and data.
- To follow and abide by this specification and any modifications made to same from time to time by Licensors. The most recent version of this specification can be obtained from a WIP Developer Integrations Consultant.
- To distribute and otherwise make available integration upgrades or modifications made by WIP to your reseller and merchant base utilizing any WIP product integrated by you or your payment application.

Any use of your point of sale system or environment to store, process or transmit cardholder data, places that point of sale system and environment within full scope of the PA-DSS. You agree to comply with PCI DSS and, if your point of sale system or environment engages in any of the above activities, to ensure that the point of sale system and environment meets all PA-DSS requirements.

By your use of these Specifications, you warrant, represent and certify your agreement to the above terms and conditions and that your payment application is compliant and adheres to the above requirements and that future versions of your payment application will continue to comply and adhere with these requirements.

Table of Contents

Document Revisions

Version	Date	Name	Change Description
1.03	07.01.2019	J Bevington	Worldpay Brand update, reformatting
1.0.2	06.08.2017	Jeff Gross	Re-branded and updated Account Updater
1.0.1	04.17.2014	Ed Plumb	Added additional Business Use Cases
1.0.0	04.03.2014	Ed Plumb	Initial document

PASS Overview

Payment Account Secure Storage (PASS) is designed to allow merchants and payment service providers alike the ability to easily comply with PCI DSS with very little time, effort, or financial impact. PASS greatly reduces an enormous financial risk inherent with storing, processing, and/or transmitting cardholder data. PASS is designed as a secure Web Service comprised of seven methods:

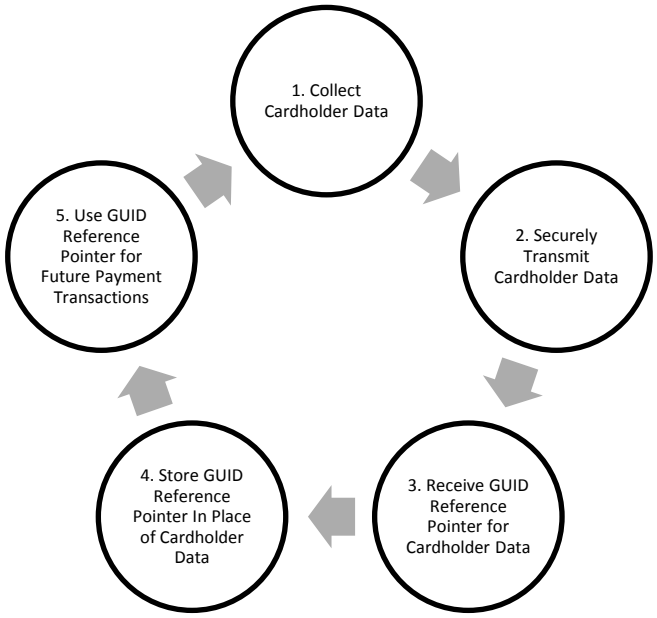
1. PaymentAccountCreate
2. PaymentAccountCreateWithTransID
3. PaymentAccountUpdate
4. PaymentAccountDelete
5. PaymentAccountQuery (see section 7.1)
6. PaymentAccountQueryRecordCount
7. PaymentAccountQueryTokenReport

These methods are used to create, update, delete, and query payment data storage accounts.

- Requests can also be sent to Express servers via the POST method in the HTTPS protocol to send XML data. XML data will then be returned in the reply.

How PASS Works

PASS follows these five simple steps:



Step 1: Collect Cardholder data
Merchants and service providers accept cardholder data as they always have; examples include, but are not limited to, track data readers (encrypted or non-encrypted) and manual data entry. Here are the options to collect card data for storage:

Direct Integration via *PaymentAccountCreate* or PaymentAccountUpdate:

- Swiped (encrypted)
- Keyed (encrypted)
- Swiped (non-encrypted device)
- Keyed (non-encrypted device)

Direct Integration via *PaymentAccountCreateWithTransID*:

- Submitted using prior TransactionID

Hosted Payments Redirect (using Express https web site for PaymentAccountCreate or PaymentAccountUpdate):

- Swiped (encrypted (USB keyboard emulation)
- Keyed (encrypted (USB keyboard emulation)
- Keyed (manually via keyboard)

Step 2: Securely Transmit Cardholder Data
The merchant's payment system transmits sensitive cardholder data over a secure 128-bit encrypted SSL connection to Element Payment Services', Inc. PCI DSS compliant storage facility.

Step 3: Receive Reference Pointer to Cardholder Data
The merchant payment system receives a globally unique identifier (GUID) as a reference to be used as a pointer to cardholder data stored remotely.

Step 4: Store Reference Pointer In Place of Cardholder Data
The Merchant stores the GUID reference pointer instead of the sensitive cardholder data. The reference pointer is of no value to hackers because the cardholder information can never be obtained by using the reference pointer. By design, extraction of full card account number is not allowed through PASS to any merchant or service provider.

Step 5: Use Reference Pointer for Future Payment Transactions
The merchant transmits the GUID reference pointer to process future payment transactions instead of transmitting the actual cardholder data. Likewise, the GUID reference pointer can be used to update, delete, and query cardholder accounts. The query functionality only returns truncated account numbers, never the entire account number.

Processing PASS Records as Payments (Transaction)

To process a transaction using PASS, the method of the transaction (e.g. *CreditCardSale* or *CheckSale*) should be called. However, instead of passing the card/account number and expiration information, the appropriate *PaymentAccountID* identifying the stored record should be submitted. The transaction will be processed in the same way as any normal transaction. Billing address and zip code information will be submitted for AVS automatically if stored in the PASS system. Otherwise, any billing address and zip code sent separately in the request will override the PASS-stored values and will be submitted for AVS.

Updating PASS Records

When performing a *PaymentAccountUpdate* to update the card number or expiration date, each of these Card object fields, along with the full address information (e.g. Address object), must be resubmitted in the update request. However, if updating only an expiration date, note that the card number is not required. When performing a *PaymentAccountUpdate* to update an Address object field only, the Card object fields are not required, but the full address details must be resubmitted in the update request. If all address details of the stored record are not known at that time, the *PaymentAccountQuery* method should be called first to collect this information. Note that the *PaymentAccountType* value cannot be updated once the record is created.

Querying PASS Records

When performing a *PaymentAccountQuery*, the search parameters include a credit card beginning and ending expiration date. This functionality will allow merchants to search any range of card expiration date information so that currently expired and/or future expired cards can be handled appropriately.

Currently Supported Payment Types

Credit Cards and Checks can be processed using PASS technology. Note that *PaymentAccountIDs* can be charged across multiple *AcceptorIDs* as long as each *AcceptorID* is set up under the same corporate *AccountID* for the merchant. However, only the original *AccountID/AcceptorID/AccountToken* combination can perform the deletions and queries.

NOTE: Every request is authenticated through use of an Element Payment Services, Inc. assigned end user AccountID and AccountToken. AccountID is a unique account identifier. AccountToken is an alpha numeric string produced using a proprietary hash algorithm.

Business Use Cases

Here are the most common business cases for merchants charging (i.e. *CreditCardSale*) or authorizing (i.e. *CreditCardAuthorize*) against a token (*PaymentAccountID*):

Card-on-File / Convenience

These transactions involve a merchant processing a payment using a token stored in the merchant point of sale application.

- This is more commonly used in a card-not-present environments (i.e. Ecommerce, Direct Marketing) to help drive customer loyalty by making the payment transaction process less cumbersome (quicker checkout). A typical card-on-file scenarios would be e-tailors (sale of goods and services through the Internet), Airline sites (i.e. Southwest.com), or Auction sites (i.e. ebay.com) for example.

Recurring

- **Ongoing Regular Amounts or Periods** – These transactions involves a merchant processing an ongoing recurring payment using a token stored in the merchant point of sale application.

  A recurring transaction is one in which a cardholder authorizes a merchant to automatically charge his other account number for the recurring or periodic delivery of goods or services. A typical recurring transaction might be an automatic bill pay for pest services, a monthly newspaper subscription, or a health club membership.

- **Installment** – These transactions involves a merchant processing an installment payment using a card number or token stored in the merchant point of sale application.

  Per Visa Best Practices Guide: "Installment transactions are often confused with recurring transactions. Visa International defines an installment transaction as: "A single purchase of goods and services billed to an account in multiple segments, over a period of time agreed between a Cardholder and Merchant."

  The distinction between the two transactions is that, a recurring transaction is payment for goods or services that are received over time, however, an installment transaction represents a single purchase, with payment occurring on a schedule agreed by a cardholder and merchant."

  A typical installment transaction might be a large order to a wholesale lumber distributor broken out into payments of the same amount for a fixed period of time. For example, a single lumber order for $10,000 paid in $2,500 increments over a 4-month time of period.

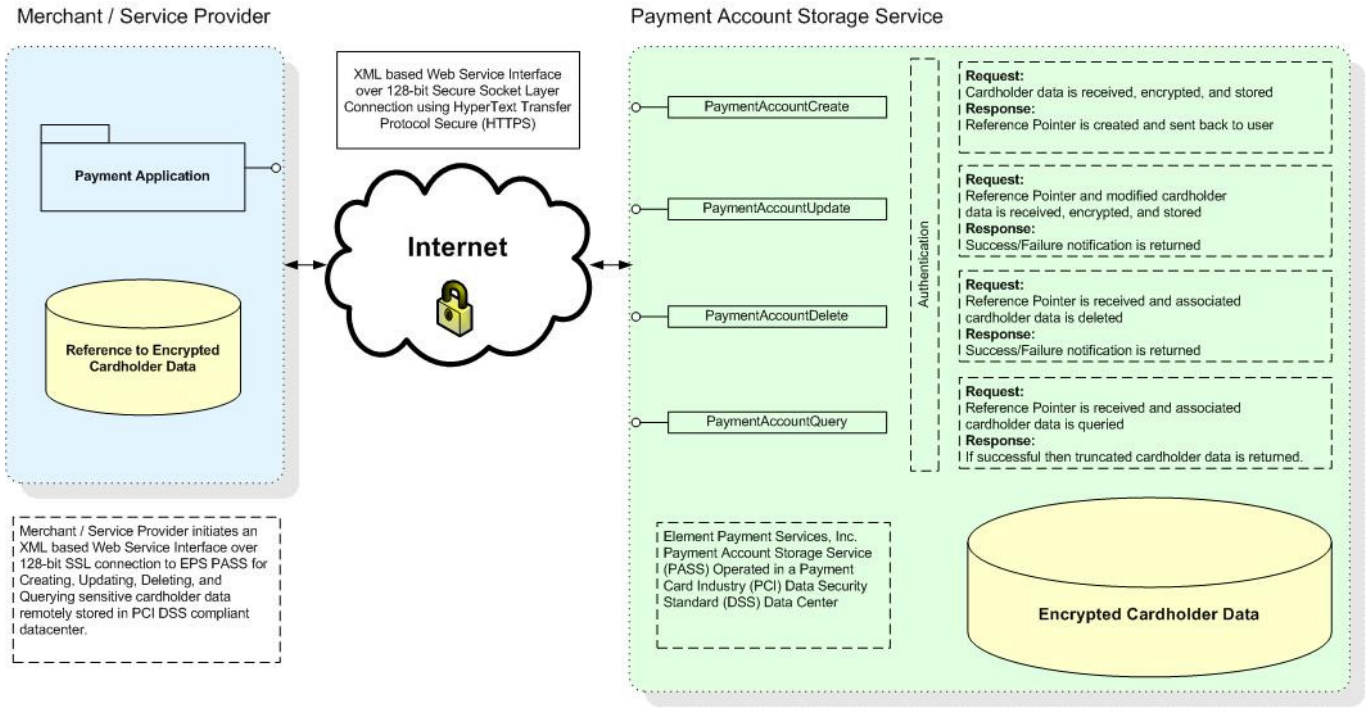
  Note: For additional details on Credentials-on-File/Card-on-file please review https://developer.vantiv.com/docs/DOC-376

Account Updater

This ensures uninterrupted service for cardholders and uninterrupted payment for merchants and is extremely beneficial in all use cases. It seamlessly updates “card on file” account information without impacting cardholders.

By automatically maintaining the accuracy of customer data, Account Updater prevents recurring billing or card-on-file disruptions due to account changes. Account Updater extends the life of automatic payment arrangements by helping secure these ongoing, revenue-generating relationships, maintaining service continuity and strengthening cardholder satisfaction.

Account Updater Diagram



Scheduled Tasks Overview

Express handles Scheduled Transactions, which is a way to set up recurring payments on an account, or a one-time-future payment. Once set up, transactions occur automatically without intervention from the merchant’s billing system.

NOTE: Scheduled tasks can be created through Element’s Virtual Terminal website or through a direct integration using *ExtendedParameters*.

Scheduling a Task (Transaction):

To schedule a transaction, call the creation method of that transaction as you would normally do (i.e. *CreditCardSale*).

- The only difference is you will also pass in the *ScheduledTask* object in the ExtendedParameters array of that method, which includes all the scheduling/frequency information. By doing this, Express will schedule the transaction.

- **NOTE: Express will only schedule the transaction, it will not execute immediately.**
- When scheduling payment transactions, you may pass a *PaymentAccountID* in for the payment information if you are already using our PASS technology for storing sensitive cardholder or payment information.
- If you choose to pass in the original card or payment information, Express will automatically create a PaymentAccount (PASS) for you and store your payment data.
- Once the Scheduled Task is created, you will be returned both a Scheduled Task ID as well as a Payment Account ID.

Scheduling Tasks with varying parameters:

You may have a scenario where you want to vary the parameters on each method call. For example: charging a cardholder a different amount each month.

- In order to accomplish this, you would create multiple Scheduled Tasks with a Run Frequency of "*OneTimeFuture*", creating one task for each occurrence.
- These tasks can be grouped together for reporting purposes by assigning the same *ScheduledTaskReferenceNumber* to all.

Editing / Updating a Scheduled Task:

The only items that you can update on a Scheduled Task are the members of the ScheduledTask class. You cannot update the actual method call parameters due to the complexity of parameters that get passed into the method. If you wish to modify those, you must delete the task and create a new one with the appropriate parameters.

Reporting on the results of Scheduled Tasks:

The *ScheduledTaskQuery* method can be used to report on the criteria in place for a particular task, as well as report on transactions that have already run based on that task.

Currently Supported Transaction Types:

CreditCardSale and *CheckSale* are the only methods that can be scheduled.

Account Updater FAQ

What is Account Updater?

The *Account Updater* service provides merchants the ability to receive automatic updates to existing records stored with our *Payment Account Secure Storage* (PASS) product to prevent billing to expired or invalid cards. When an expired or invalid card is flagged, our system will query the payment brands (Visa and MasterCard) for updated cardholder information. If the information is available, your customer's account on file (PASS billing record) will be updated and available for future billing.

How does Account Updater work?

Merchant locations must first enroll in Worldpay's *Account Updater* service, which is designed to work through a software application integration to the *Express* platform. Software applications can be set to flag PASS records for update each time a PASS transaction is declined, and can also allow merchants to manually flag for update any PASS records of their choosing. At the end of the day, all cards that have been flagged will be submitted to the *Account Updater* service. Software applications can query

flagged PASS records to determine if a card has been successfully updated. Updates are usually received within a couple business days (but could take up to approximately twelve business days), and we will not update a PASS record until an update response is received. If the issuer provides no update information after approximately twelve days, the *Express* platform will set the token status as "NoResponse" (enum value 10). If an update was successful, merchants can submit the updated PASS record for future billing. Note that if the software application stores any non-sensitive, truncated card information, then that truncated information may also need to be updated to reflect the updated PASS record on file.

Which card types can be flagged for update?

Only Visa and MasterCard cards can be enrolled in the *Account Updater* service.

Which transaction declines should be flagged for update?

If manually flagging records, it is recommended that the following *ExpressResponseCode* decline values be flagged for update:

RC	Type	Description
20	Decline	Card issuer has declined the transaction.
21	Expired Card	Card is expired.
24	Pick Up Card	Card issuer has declined the transaction and wants to recover the card.

Which updates are considered valid updates?

The following *PASSUpdaterStatus* response values resulting from a *PaymentAccountQuery* request are considered valid updates:

Match Type	Description
Match: Account Change	Account match was found and account number has changed. The PASS record has been updated with the new credit card number.
Match: Expiration Change	Account match was found and expiration date has changed. The PASS record has been updated with the new expiration date.
Match: Account Closed	Account match was found. Account has been closed. The PASS record will not be updated. This allows a merchant to identify PASS records that will never produce an approval by distinguishing between declines that are legitimate (due to closed accounts) and those resulting because the cardholder is too close to their credit limit to use the card on file.
Match: Contact Cardholder	Account match was found. The PASS record will not be updated. The merchant will need to contact the cardholder to inquire about the status of the account. This information allows a merchant to identify PASS records that will never produce an approval by distinguishing between declines that are legitimate (due to closed accounts) and those resulting because the cardholder is too close to the credit limit to use the card on file.

Integration to the Express platform

- To flag an existing PASS record for a **one-time manual update** (recommended after specified declines), the software application can initiate a *PaymentAccountAutoUpdate* request using a *PASSUpdaterBatchStatus* of "IncludedInNextBatch".

- To flag an existing PASS record to be **updated automatically** after any future specified decline, the *PASSUpdaterOption* should be set to "AutoUpdateEnabled".

- **For direct integrations** (where *PaymentAccountCreate* is submitted directly from the software application), software applications can also flag a PASS record for a one-time manual update by submitting the *PASSUpdaterBatchStatus* of "IncludedInNextBatch" in the *PaymentAccountCreate* request.

- To determine the **current status of a previously-flagged record**, the software can initiate a *PaymentAccountQuery* request using *PASSUpdaterStatus*, *PASSUpdaterBatchStatus*, *PASSUpdaterDateTimeBegin*, *PASSUpdaterDateTimeEnd*, or any other existing query parameter (such as *PaymentAccountID*, *PaymentAccountReferenceNumber*, *Expiration Date*, *PaymentBrand*, etc.) A full list of possible *PASSUpdaterStatus* update responses are included below.

  Note that when initially flagged for update, the *PASSUpdaterStatus* will initially be set to "NotUpdated" (enum value 14) until the card number is submitted in the nightly update request file at Vantiv. Once that happens, the *PASSUpdaterStatus* will be set to "UpdateInProgress" (enum value 1) until an update response is received.

Match Type	Enum	Description
Null	0	N/A
Update in progress	1	Update in progress.
Match: No Changes	2	Account match was found, but there are no changes.
Match: Account Change	3	Account match was found and account number has changed. Account number has been updated.
Match: Expiration Change	4	Account match was found and expiration date has changed. Expiration date has been updated.
Match: Account Closed	5	Account match was found. Account has been closed.
Match: Contact Cardholder	6	Account match was found. Contact the cardholder to update the account information.
No Match: Participating	7	Account match not found on participating BIN.
No Match: Non-Participating	8	Account match not found on non-participating BIN.
Invalid Info	9	Invalid information. e.g. Invalid card number.
No Response	10	No response. e.g. Request submitted, but no response.
Not Allowed	11	Not allowed. e.g. Merchant may not be set up for Account Updater service.
Error	12	Error.
PASS Updater Disabled	13	PASS Updater is disabled.
Not Updated	14	Not updated.

- If you wish to disable the ability for a currently-flagged record to be updated, the software application can submit a *PaymentAccountAutoUpdate* request using a *PASSUpdaterOption* of "*AutoUpdateDisabled*" and a *PASSUpdaterBatchStatus* of "NotIncludedInNextBatch".

Additional Account Updater Notes:

- As long as the merchant is enrolled in the service, the *PaymentAccountCreate* looks at the *PASSUpdaterBatchStatus* (e.g. "IncludedInNextBatch") only. *PaymentAccountCreate* will ignore the PASSUpdaterOption (e.g. "Disable").
- On all *PaymentAccountAutoUpdate* requests, the *PASSUpdaterOption* of "AutoUpdateDisabled" overrides any other existing status or value.
- *PASSUpdaterDate* is the date that the record was updated in PASS (from an account updater process).
- When a PASS record has actually been submitted to the acquirer for an update (this happens once per night in *Express*), the PASS record *PASSUpdaterStatus* field will go into an "UpdateInProgress" status (enum value **1**) and the *PASSUpdaterBatchStatus* field will go into an "IncludedInNextBatch" status (enum value **1**) until the update process is complete within *Express* (e.g. up to ~12 days later).

  NOTE: If a PASS record is deleted (*PaymentAccountDelete*) while in an "In Progress" status, that PASS record Account Update detail will never be returned to the software application or merchant.

  NOTE: If a PASS record is updated (*PaymentAccountUpdate*) while in an "In Progress status", that PASS record Account Update detail may override any details modified using the *PaymentAccountUpdate* method during an "In Progress" status. In this scenario, it is recommended that merchants not be allowed to modify the card number or expiration date on file until the Auto Update process is completed. As a work-around, merchants wishing to update the card number or expiration date of a record currently In Progress can delete that existing record (*PaymentAccountDelete*) and submit the full cardholder data again via the *PaymentAccountCreate* method.
- Once the update process is complete within *Express* (e.g. up to ~12 days later), the *PASSUpdaterStatus* of the PASS record will be set appropriately (depending on the update response) and the *PASSUpdaterBatchStatus* will immediately be set to "*NotIncludedInNextBatch*" (enum value **2**). The *PASSUpdaterStatus* value will not change unless the PASS record is again flagged for an update (at which point it will change back to "UpdateInProgress" (enum value **1**) when the nightly update file is submitted to the acquirer).

PASS Methods

PaymentAccountCreate

The *PaymentAccountCreate* method is used to create a new Payment Account record. The required fields are dependent upon the PaymentAccountType selected.

Input Fields

Name	Class	Required	Description
ApplicationID	Application	Required	Unique application identifier
ApplicationName	Application	Required	Name of application
ApplicationVersion	Application	Required	Version of application
AccountID	Credentials	Required	Unique account identifier
AccountToken	Credentials	Required	Secret token used for authentication

AcceptorID	Credentials	Required	MerchantID
PaymentAccountType	PaymentAccount	Required	Payment Account Type
PaymentAccountReferenceNumber	PaymentAccount	Required	Payment Account Reference Number
PASSUpdaterBatchStatus	PaymentAccount	Optional [2]	Specifies whether or not the PASS record will be included in the next PASS Updater batch
PASSUpdaterOption	PaymentAccount	Optional [2]	Specifies whether or not the PASS record will be updated automatically
CardNumber	Card	Conditional [1]	Card Number
ExpirationMonth	Card	Conditional [1]	Expiration Month (MM)
ExpirationYear	Card	Conditional [1]	Expiration Year (YY)
Track1Data	Card	Conditional [1]	Track1 Data
Track2Data	Card	Conditional [1]	Track2 Data
MagneprintData	Card	Conditional [1]	Magneprint Data
EncryptedTrack1Data	Card	Conditional [1]	Encrypted Track 1 Data from integrated encryption device
EncryptedTrack2Data	Card	Conditional [1]	Encrypted Track 2 Data from integrated encryption device
EncryptedCardData	Card	Conditional [1]	Encrypted Card Data (keyed) from integrated encryption device
CardDataKeySerialNumber	Card	Conditional [1]	Card Data Key Serial Number from integrated encryption device
EncryptedFormat	Card	Optional	Encryption format from integrated encryption device
AccountNumber	DemandDepositAccount	Conditional - (Required if PaymentAccountType is a DDA account)	Account Number
RoutingNumber	DemandDepositAccount	Conditional - (Required if PaymentAccountType is a DDA account)	Routing Number
BillingName	Address	Optional	The name used for billing purposes
BillingAddress1	Address	Optional	The street address used for billing purposes
BillingAddress2	Address	Optional	The street address used for billing purposes
BillingCity	Address	Optional	The city name used for billing purposes
BillingState	Address	Optional	The state name used for billing purposes
BillingZipcode	Address	Optional	The zip code used for billing purposes
BillingEmail	Address	Optional	The email address used for billing purposes
BillingPhone	Address	Optional	The phone number used for billing purposes
ShippingName	Address	Optional	The name used for shipping purposes
ShippingAddress1	Address	Optional	The street address used for Shipping purposes
ShippingAddress2	Address	Optional	The street address used for Shipping purposes
ShippingCity	Address	Optional	The city name used for Shipping purposes
ShippingState	Address	Optional	The state name used for Shipping purposes

ShippingZipcode	Address	Optional	The zip code used for Shipping purposes
ShippingEmail	Address	Optional	The email address used for Shipping purposes
ShippingPhone	Address	Optional	The phone number used for Shipping purposes

1	If PaymentAccountType = CreditCard then exactly one of the following field groups needs to be included: CardNumber / ExpirationMonth / ExpirationYear **or** Track1Data **or** Track2Data **or** MagneprintData **or** EncryptedTrack2Data **or** EncryptedTrack1Data **or** EncryptedCardData
2	Used with Account Updater service only.

Output Fields

Name	Class	Returned	Description
ExpressResponseCode	Response	Returned	Express Response Code
ExpressResponseMessage	Response	Returned	Express Response Message
ExpressTransactionDate	Response	Returned	Express transaction date formatted [YYYYMMDD]
ExpressTransactionTime	Response	Returned	Express transaction time formatted [HHMMSS]
ExpressTransactionTimezone	Response	Returned	Express transaction UTC time zone
ServicesID	Response	Returned	Unique services identifier
PaymentAccountID	Response.PaymentAccount	Returned	Unique GUID that identifies the Payment Account.
PaymentAccountReferenceNumber	Response.PaymentAccount	Returned	Payment Account Reference Number

PaymentAccountDelete

The *PaymentAccountDelete* method is used to delete a Payment Account record.

- This method will mark a Payment Account record for deletion and will be purged from the system within twenty-four hours.

Input Fields

Name	Class	Required	Description
ApplicationID	Application	Required	Unique application identifier
ApplicationName	Application	Required	Name of application
ApplicationVersion	Application	Required	Version of application
AccountID	Credentials	Required	Unique account identifier
AccountToken	Credentials	Required	Secret token used for authentication
AcceptorID	Credentials	Required	MerchantID
PaymentAccountID	PaymentAccount	Required	Unique GUID that identifies the Payment Account.

Output Fields

Name	Class	Returned	Description
ExpressResponseCode	Response	Returned	Express Response Code

ExpressResponseMessage	Response	Returned	Express Response Message
ExpressTransactionDate	Response	Returned	Express transaction date formatted [YYYYMMDD]
ExpressTransactionTime	Response	Returned	Express transaction time formatted [HHMMSS]
ExpressTransactionTimezone	Response	Returned	Express transaction UTC time zone
ServicesID	Response	Returned	Unique services identifier

PaymentAccountUpdate

The *PaymentAccountUpdate* method is used to update an existing Payment Account record.

- The required fields are dependent upon the PaymentAccountType selected.

Input Fields

Name	Class	Required	Description
ApplicationID	Application	Required	Unique application identifier
ApplicationName	Application	Required	Name of application
ApplicationVersion	Application	Required	Version of application
AccountID	Credentials	Required	Unique account identifier
AccountToken	Credentials	Required	Secret token used for authentication
AcceptorID	Credentials	Required	MerchantID
PaymentAccountID	PaymentAccount	Required	Unique GUID that identifies the Payment Account.
PaymentAccountReferenceNumber	PaymentAccount	Required	Payment Account Reference Number
PaymentAccountType	PaymentAccount	Required [1]	Payment Account Type
Track1Data	Card	Conditional [2]	Track1 Data
Track2Data	Card	Conditional [2]	Track2 Data
MagneprintData	Card	Conditional [2]	Magneprint Data
CardNumber	Card	Optional	Card Number
ExpirationMonth	Card	Conditional - (Required if PaymentAccountType is CreditCard)	Expiration Month (MM)
ExpirationYear	Card	Conditional - (Required if PaymentAccountType is CreditCard)	Expiration Year (YY)
EncryptedTrack1Data	Card	Conditional [2]	Encrypted Track 1 Data from integrated encryption device
EncryptedTrack2Data	Card	Conditional [2]	Encrypted Track 2 Data from integrated encryption device
EncryptedCardData	Card	Conditional [2]	Encrypted Card Data (keyed) from integrated encryption device
CardDataKeySerialNumber	Card	Conditional [2]	Card Data Key Serial Number from integrated encryption device
EncryptedFormat	Card	Optional	Encryption format from integrated encryption device

AccountNumber	DemandDepositAccount	Conditional - (Required if PaymentAccountType is a DDA account)	Account Number
RoutingNumber	DemandDepositAccount	Conditional - (Required if PaymentAccountType is a DDA account)	Routing Number
BillingName	Address	Optional	The name used for billing purposes
BillingAddress1	Address	Optional	The street address used for billing purposes
BillingAddress2	Address	Optional	The street address used for billing purposes
BillingCity	Address	Optional	The city name used for billing purposes
BillingState	Address	Optional	The state name used for billing purposes
BillingZipcode	Address	Optional	The zip code used for billing purposes
BillingEmail	Address	Optional	The email address used for billing purposes
BillingPhone	Address	Optional	The phone number used for billing purposes
ShippingName	Address	Optional	The name used for shipping purposes
ShippingAddress1	Address	Optional	The street address used for Shipping purposes
ShippingAddress2	Address	Optional	The street address used for Shipping purposes
ShippingCity	Address	Optional	The city name used for Shipping purposes
ShippingState	Address	Optional	The state name used for Shipping purposes
ShippingZipcode	Address	Optional	The zip code used for Shipping purposes
ShippingEmail	Address	Optional	The email address used for Shipping purposes
ShippingPhone	Address	Optional	The phone number used for Shipping purposes

1	PaymentAccountType cannot be updated
2	If PaymentAccountType = CreditCard then exactly one of the following field groups needs to be included: CardNumber / ExpirationMonth / ExpirationYear **or** Track1Data **or** Track2Data **or** MagneprintData **or** EncryptedTrack2Data **or** EncryptedTrack1Data **or** EncryptedCardData

Output Fields

Name	Class	Returned	Description
ExpressResponseCode	Response	Returned	Express Response Code
ExpressResponseMessage	Response	Returned	Express Response Message
ExpressTransactionDate	Response	Returned	Express transaction date formatted [YYYYMMDD]
ExpressTransactionTime	Response	Returned	Express transaction time formatted [HHMMSS]
ExpressTransactionTimezone	Response	Returned	Express transaction UTC time zone
ServicesID	Response	Returned	Unique services identifier
PaymentAccountID	Response.PaymentAccount	Returned	Unique GUID that identifies the Payment Account.

PaymentAccountReferenceNumber	Response.PaymentAccount	Returned	Payment Account Reference Number

PaymentAccountQuery

The *PaymentAccountQuery* method is used to query existing Payment Account record(s). The *QueryData* field will return Payment Account information in XML format, with a maximum of 1000 records returned.

- This method will return the resulting query in xml format based on the input parameters passed in via the *PaymentAccountParameters* object.
- At least one *PaymentAccountParameters* value required to query records

Input Fields

Name	Class	Required	Description
ApplicationID	Application	Required	Unique application identifier
ApplicationName	Application	Required	Name of application
ApplicationVersion	Application	Required	Version of application
AccountID	Credentials	Required	Unique account identifier
AccountToken	Credentials	Required	Secret token used for authentication
AcceptorID	Credentials	Required	MerchantID
PaymentAccountID	PaymentAccountParameters	Optional	Unique GUID that identifies the Payment Account.
PaymentAccountReferenceNumber	PaymentAccountParameters	Optional	Payment Account Reference Number
PaymentBrand	PaymentAccountParameters	Optional	Card type query value
ExpirationMonthBegin	PaymentAccountParameters	Optional	Card expiration month query value
ExpirationMonthEnd	PaymentAccountParameters	Optional	Card expiration month query value
ExpirationYearBegin	PaymentAccountParameters	Optional	Card expiration year search value
ExpirationYearEnd	PaymentAccountParameters	Optional	Card expiration year query value
TransactionSetupID	PaymentAccountParameters	Optional	Transaction Setup IDs
PASSUpdaterDateTimeBegin	PaymentAccountParameters	Optional	Begin date/time of range formatted [yyyy-MM-dd HH:mm:ss.fff]
PASSUpdaterDateTimeEnd	PaymentAccountParameters	Optional	End date/time of range formatted [yyyy-MM-dd HH:mm:ss.fff]
PASSUpdaterBatchStatus	PaymentAccountParameters	Optional	Specifies whether or not the PASS record will be included in the next PASS Updater batch
PASSUpdaterStatus	PaymentAccountParameters	Optional	Specifies the match status of the PASS record

Output Fields

Name	Class	Returned	Description
ExpressResponseCode	Response	Returned	Express Response Code
ExpressResponseMessage	Response	Returned	Express Response Message
ExpressTransactionDate	Response	Returned	Express transaction date formatted [YYYYMMDD]
ExpressTransactionTime	Response	Returned	Express transaction time formatted [HHMMSS]
ExpressTransactionTimezone	Response	Returned	Express transaction UTC time zone

QueryData	Response	Returned	Records returned in xml format (<Items><Item>…</Item>…</Items>). A maximum of 1000 records can be returned.
ServicesID	Response	Returned	Unique services identifier
PaymentAccountID	Response.PaymentAccount	Returned	Unique GUID that identifies the Payment Account.
PaymentAccountReferenceNumber	Response.PaymentAccount	Conditional Returned if input value passed	Payment Account Reference Number

PaymentAccountAutoUpdate

The *PaymentAccountAutoUpdate* method is used only with the PASS Auto Updater product.

Input Fields

Name	Class	Required	Description
ApplicationID	Application	Required	Unique application identifier
ApplicationName	Application	Required	Name of application
ApplicationVersion	Application	Required	Version of application
AccountID	Credentials	Required	Unique account identifier
AccountToken	Credentials	Required	Secret token used for authentication
AcceptorID	Credentials	Required	MerchantID
PaymentAccountID	PaymentAccount	Required	Unique GUID that identifies the Payment Account.
PASSUpdaterBatchStatus	PaymentAccount	Required	Specifies whether or not the PASS record will be included in the next PASS Updater batch
PASSUpdaterOption	PaymentAccount	Optional	Specifies whether or not the PASS record will be updated automatically

Output Fields

Name	Class	Returned	Description
ExpressResponseCode	Response	Returned	Express Response Code
ExpressResponseMessage	Response	Returned	Express Response Message
ExpressTransactionDate	Response	Returned	Express transaction date formatted [YYYYMMDD]
ExpressTransactionTime	Response	Returned	Express transaction time formatted [HHMMSS]
ExpressTransactionTimezone	Response	Returned	Express transaction UTC time zone
ServicesID	Response	Returned	Unique services identifier
PaymentAccountID	Response. PaymentAccount	Returned	Unique GUID that identifies the Payment Account.

PaymentAccountCreateWithTransID

The *PaymentAccountCreateWithTransID* method is used to create a new Payment Account record from an existing TransactionID.

- It is recommended that *TransactionID*'s only from the following Transaction Types be used to generate a *PaymentAccountID: CreditCardSale, CreditCardCredit, CreditCardAuthorization, CreditCardAVSOnly,*

CreditCardForce, CreditCardBalanceInquiry, DebitCardSale, DebitCardReturn, CheckSale, CheckCredit, CheckVerification.

- The *PaymentAccountType* submitted must be correct for the associated *TransactionID* submitted.
- The *PaymentAccountCreateWithTransID* can be used to create Payment Account records from transactions as old as the Element Data Retention policy allows (45 days)

Input Fields

Name	Class	Required	Description
ApplicationID	Application	Required	Unique application identifier
ApplicationName	Application	Required	Name of application
ApplicationVersion	Application	Required	Version of application
AccountID	Credentials	Required	Unique account identifier
AccountToken	Credentials	Required	Secret token used for authentication
AcceptorID	Credentials	Required	MerchantID
PaymentAccountType	PaymentAccount	Required	Payment Account Type
PaymentAccountReferenceNumber	PaymentAccount	Required	Payment Account Reference Number
TransactionID	Transaction	Required	Unique Transaction identifier
BillingName	Address	Optional	The name used for billing purposes
BillingAddress1	Address	Optional	The street address used for billing purposes
BillingAddress2	Address	Optional	The street address used for billing purposes
BillingCity	Address	Optional	The city name used for billing purposes
BillingState	Address	Optional	The state name used for billing purposes
BillingZipcode	Address	Optional	The zip code used for billing purposes
BillingEmail	Address	Optional	The email address used for billing purposes
BillingPhone	Address	Optional	The phone number used for billing purposes
ShippingName	Address	Optional	The name used for shipping purposes
ShippingAddress1	Address	Optional	The street address used for Shipping purposes
ShippingAddress2	Address	Optional	The street address used for Shipping purposes
ShippingCity	Address	Optional	The city name used for Shipping purposes
ShippingState	Address	Optional	The state name used for Shipping purposes
ShippingZipcode	Address	Optional	The zip code used for Shipping purposes
ShippingEmail	Address	Optional	The email address used for Shipping purposes
ShippingPhone	Address	Optional	The phone number used for Shipping purposes

Output Fields

Name	Class	Returned	Description

ExpressResponseCode	Response	Returned	Express Response Code
ExpressResponseMessage	Response	Returned	Express Response Message
ExpressTransactionDate	Response	Returned	Express transaction date formatted [YYYYMMDD]
ExpressTransactionTime	Response	Returned	Express transaction time formatted [HHMMSS]
ExpressTransactionTimezone	Response	Returned	Express transaction UTC time zone
ServicesID	Response	Returned	Unique services identifier
PaymentAccountID	Response.PaymentAccount	Returned	Unique GUID that identifies the Payment Account.
PaymentAccountReferenceNumber	Response.PaymentAccount	Returned	Payment Account Reference Number

PaymentAccountQueryRecordCount

The *PaymentAccountQueryRecordCount* method is used to query the number of existing Payment Account record(s).

- The QueryData field will return Payment Account record count and paging information in xml format. Paging details can be used with the *PaymentAccountQueryTokenReport* method.
- This method will return the resulting query in xml format.

Input Fields

Name	Class	Required	Description
ApplicationID	Application	Required	Unique application identifier
ApplicationName	Application	Required	Name of application
ApplicationVersion	Application	Required	Version of application
AccountID	Credentials	Required	Unique account identifier
AccountToken	Credentials	Required	Secret token used for authentication
AcceptorID	Credentials	Required	MerchantID

Output Fields

Name	Class	Returned	Description
ExpressResponseCode	Response	Returned	Express Response Code
ExpressResponseMessage	Response	Returned	Express Response Message
ExpressTransactionDate	Response	Returned	Express transaction date formatted [YYYYMMDD]
ExpressTransactionTime	Response	Returned	Express transaction time formatted [HHMMSS]
ExpressTransactionTimezone	Response	Returned	Express transaction UTC time zone
ServicesID	Response	Returned	Unique services identifier

QueryData	Response	Returned	Records returned in xml format (<QueryData><Results>…</Results>…</Results>).
			Within <Results>…</Results>:
			RecordCount = # PASS Records
			PageCount = # of Pages

PaymentAccountQueryTokenReport

The *PaymentAccountQueryTokenReport* method is used to obtain details about existing Payment Account records. The QueryData field will return Payment Account record information in xml format. This method will return the resulting query in xml format.

Input Fields

Name	Class	Required	Description
ApplicationID	Application	Required	Unique application identifier
ApplicationName	Application	Required	Name of application
ApplicationVersion	Application	Required	Version of application
AccountID	Credentials	Required	Unique account identifier
AccountToken	Credentials	Required	Secret token used for authentication
AcceptorID	Credentials	Required	MerchantID
Page	Paging	Required	Page number of records to retrieve, e.g. 1

Output Fields

Name	Class	Returned	Description
ExpressResponseCode	Response	Returned	Express Response Code
ExpressResponseMessage	Response	Returned	Express Response Message
ExpressTransactionDate	Response	Returned	Express transaction date formatted [YYYYMMDD]
ExpressTransactionTime	Response	Returned	Express transaction time formatted [HHMMSS]
ExpressTransactionTimezone	Response	Returned	Express transaction UTC time zone
ServicesID	Response	Returned	Unique services identifier
QueryData	Response	Returned	Records returned in xml format (<Items><I>…</I>…</Items>). Within <I>…</I>: <T>…</T> = PaymentAccountID <R>…</R> = PaymentAccount ReferenceNumber

ScheduledTaskDelete

The *ScheduledTaskDelete* method is used to delete an existing Scheduled Task. This method will delete a Scheduled Task permanently. NOTE: Accessible through the Express Services Interface

Input Fields

Name	Class	Required	Description
ApplicationID	Application	Required	Unique application identifier
ApplicationName	Application	Required	Name of application
ApplicationVersion	Application	Required	Version of application
AccountID	Credentials	Required	Unique account identifier
AccountToken	Credentials	Required	Secret token used for authentication

AcceptorID	Credentials	Required	MerchantID
ScheduledTaskID	ScheduledTask	Required	Unique value that identifies the Scheduled Task.

Output Fields

Name	Class	Returned	Description
ExpressResponseCode	Response	Returned	Express Response Code
ExpressResponseMessage	Response	Returned	Express Response Message
ExpressTransactionDate	Response	Returned	Express transaction date formatted [YYYYMMDD]
ExpressTransactionTime	Response	Returned	Express transaction time formatted [HHMMSS]
ExpressTransactionTimeZone	Response	Returned	Express transaction UTC time zone

ScheduledTaskQuery

The *ScheduledTaskQuery* method is used to query existing Scheduled Tasks. The QueryData field will return Scheduled Task information in xml format. Note: Accessible through the Express Services Interface.

- This method will return the resulting query in xml format based on the input parameters passed in via the *ScheduledTaskParameters* object.
- At least one *ScheduledTaskParameters* value required to query records

Input Fields

Name	Class	Required	Description
ApplicationID	Application	Required	Unique application identifier
ApplicationName	Application	Required	Name of application
ApplicationVersion	Application	Required	Version of application
AccountID	Credentials	Required	Unique account identifier
AccountToken	Credentials	Required	Secret token used for authentication
AcceptorID	Credentials	Required	MerchantID
ScheduledTaskID	ScheduledTaskParameters	Optional	Unique GUID that identifies the Scheduled Task.
ScheduledTaskReferenceNumber	ScheduledTaskParameters	Optional	User defined unique reference number that identifies this scheduled task.
ScheduledTaskName	ScheduledTaskParameters	Optional	User generated name of the task.
ScheduledTaskGroupID	ScheduledTaskParameters	Optional	Express platform generates this.
RunStartDate	ScheduledTaskParameters	Optional	The start date. Format is YYYYMMDD.
RunUntilCancelFlag	ScheduledTaskParameters	Optional	If this is set to true, the task will run every scheduled time until the task is deleted by the user.
RunFrequency	ScheduledTaskParameters	2	Specifies how often an event should occur.
ScheduledTaskStatus	ScheduledTaskParameters	2	The status of the task. You can set this to Active or Disabled.
ScheduledTaskQueryType	ScheduledTaskParameters	Optional	Query type used to provide additional event information (0=Default, 1=RunLog)
ScheduledTaskRunStatus	ScheduledTaskParameters	Optional	The run status of the task.

Output Fields

Name	Class	Returned	Description
ExpressResponseCode	Response	Returned	Express Response Code
ExpressResponseMessage	Response	Returned	Express Response Message
ExpressTransactionDate	Response	Returned	Express transaction date formatted [YYYYMMDD]
ExpressTransactionTime	Response	Returned	Express transaction time formatted [HHMMSS]
ExpressTransactionTimeZone	Response	Returned	Express transaction UTC time zone
QueryData	Response	Returned	Records returned in xml format (<Items><Item>…</Item>…</Items>). A maximum of 1000 records can be returned.
ServicesID	Response	Returned	Unique services identifier
ScheduledTaskID	Response.ScheduledTask	Returned	Unique value that identifies the Scheduled Task
ScheduledTaskReferenceNumber	Response.ScheduledTask	Conditional Returned if input value passed	User defined unique reference number that identifies this scheduled task.

ScheduledTaskUpdate

The *ScheduledTaskUpdate* method is used to update an existing Scheduled Task. Note: Accessible through the Express Services Interface. This method will update a Scheduled Task

Input Fields

Name	Class	Required	Description
ApplicationID	Application	Required	Unique application identifier
ApplicationName	Application	Required	Name of application
ApplicationVersion	Application	Required	Version of application
AccountID	Credentials	Required	Unique account identifier
AccountToken	Credentials	Required	Secret token used for authentication
AcceptorID	Credentials	Required	MerchantID
ScheduledTaskID	ScheduledTask	Required	Unique value that identifies the Scheduled Task.
ScheduledTaskReferenceNumber	ScheduledTask	Required	
RunFrequency	ScheduledTask	Required	
RunUntilCancelFlag	ScheduledTask	Required	
RunCycles	ScheduledTask	Required only when RunUntilCancelFlag set to false	
ScheduledTaskStatus	ScheduledTask	Required	
ScheduledTaskName	ScheduledTask	Optional	User generated name of the task.

ScheduledTaskGroupID	ScheduledTask	Optional	Express platform generates this.

Output Fields

Name	Class	Returned	Description
ExpressResponseCode	Response	Returned	Express Response Code
ExpressResponseMessage	Response	Returned	Express Response Message
ExpressTransactionDate	Response	Returned	Express transaction date formatted [YYYYMMDD]
ExpressTransactionTime	Response	Returned	Express transaction time formatted [HHMMSS]
ExpressTransactionTimeZone	Response	Returned	Express transaction UTC time zone

ScheduledTaskRetry

The *ScheduledTaskRetry* method is used to rerun a single, previously non-approved Scheduled Task. Note: Accessible through the Express Services Interface.

Input Fields

Name	Class	Required	Description
ApplicationID	Application	Required	Unique application identifier
ApplicationName	Application	Required	Name of application
ApplicationVersion	Application	Required	Version of application
AccountID	Credentials	Required	Unique account identifier
AccountToken	Credentials	Required	Secret token used for authentication
AcceptorID	Credentials	Required	MerchantID
ScheduledTaskRunLogID	ScheduledTask	Optional	Run log ID of a Scheduled Task

Output Fields

Name	Class	Returned	Description
ExpressResponseCode	Response	Returned	Express Response Code
ExpressResponseMessage	Response	Returned	Express Response Message
ExpressTransactionDate	Response	Returned	Express transaction date formatted [YYYYMMDD]
ExpressTransactionTime	Response	Returned	Express transaction time formatted [HHMMSS]
ExpressTransactionTimeZone	Response	Returned	Express transaction UTC time zone
ServicesID	Response	Returned	Unique services identifier
TransactionID	Response.Transaction	Returned	Unique transaction identifier
ProcessorName	Response.Transaction	Returned	Name of processor
TransactionStatus	Response.Transaction	Returned	Description of status/state of transaction
TransactionStatusCode	Response.Transaction	Returned	Status/State of transaction
ScheduledTaskID	Response.ScheduledTask	Returned	Unique GUID that identifies the Payment Account.

PASS Data Export Policy

The PASS Data Export is only available to those clients that have terminated their PASS Agreement or Merchant Agreement with Element. For cost details associated with the PASS Data Export, please refer to your existing PASS Agreement.

PASS Data Export Options

1. PASS Data Export to be provided directly to a Merchant
2. PASS Data Export to be provided directly to a Service Provider (PCI-DSS or PA-DSS)

Whether the data is to be provided to a Merchant or Service Provider, the organization accepting the data from Element will be required to be PCI-DSS or PA-DSS compliant. PASS Data Export Standard Procedures (Merchant or Service Provider).

Step #1:
If the PASS Data Export is to be provided to a Merchant, that Merchant must be validated by a third-party as a PCI-DSS compliant Merchant, which requires an On-Site PCI Data Security Assessment by a Qualified Security Assessor (QSA). Self-Assessments will not be accepted. A list of PCI-approved QSA companies is available from the PCI Security Standards Council Web Site at https://www.pcisecuritystandards.org/approved_companies_providers/qsa_companies.php

If the PASS Data Export is to be provided to a Service Provider, that Service Provider must be validated by a third-party as a PCI-DSS compliant Level 1 Service Provider or as a PA-DSS compliant Payment Application, both of which require an On-Site PCI Data Security Assessment by a Qualified Security Assessor (QSA or PA-QSA). A list of PCI-approved QSA/PA-QSA companies is available from the PCI Security Standards Council Web Site at https://www.pcisecuritystandards.org/approved_companies_providers/qsa_companies.php and https://www.pcisecuritystandards.org/approved_companies_providers/payment_application_qsas.php

Step #2:
Once the On-Site PCI Data Security Assessment has been completed, to verify compliance, Element will require a copy of the Merchant or Service Provider PCI Certificate of Validation from the QSA (or the PA-DSS acceptance letter from the PA-QSA).

Step #3:
Once PCI compliance has been verified, Element will provide an Export Addendum to the PASS Agreement that the merchant is required to sign. This addendum will outline the export process, including information regarding the data elements to be supplied, destination of the export, and specific data transfer details.

Step #4:
Once the Export Addendum to the PASS Agreement has been signed, Element will provide the Merchant with the PASS Data Export cost details based on the number of PASS records currently on file for the Merchant.

Step #5:

Once the Merchant has paid Express for the cost associated with the data transfer project, the PASS Data Export process will begin.