electronic payment exchange

API - Batch File

Public - Security Level 0

March 2022

REVISION HISTORY

Date	Version	Author(s)	Comments
5/20/16	2.0	C. Meaney	Reformatting
10/27/16	2.1	C. Meaney	Clarified FILE_ MODIFIER description.
3/23/17	2.2	C. Meaney	Minor edits to Example Response File.

CONFIDENTIALITY STATEMENT

This document contains confidential and proprietary information that belongs exclusively to Electronic Payment Exchange (EPX). Receipt of this document imposes the obligation on the recipient to protect the information from loss or disclosure to other parties.

This publication may not be reproduced or distributed for any purpose without the written permission of EPX.

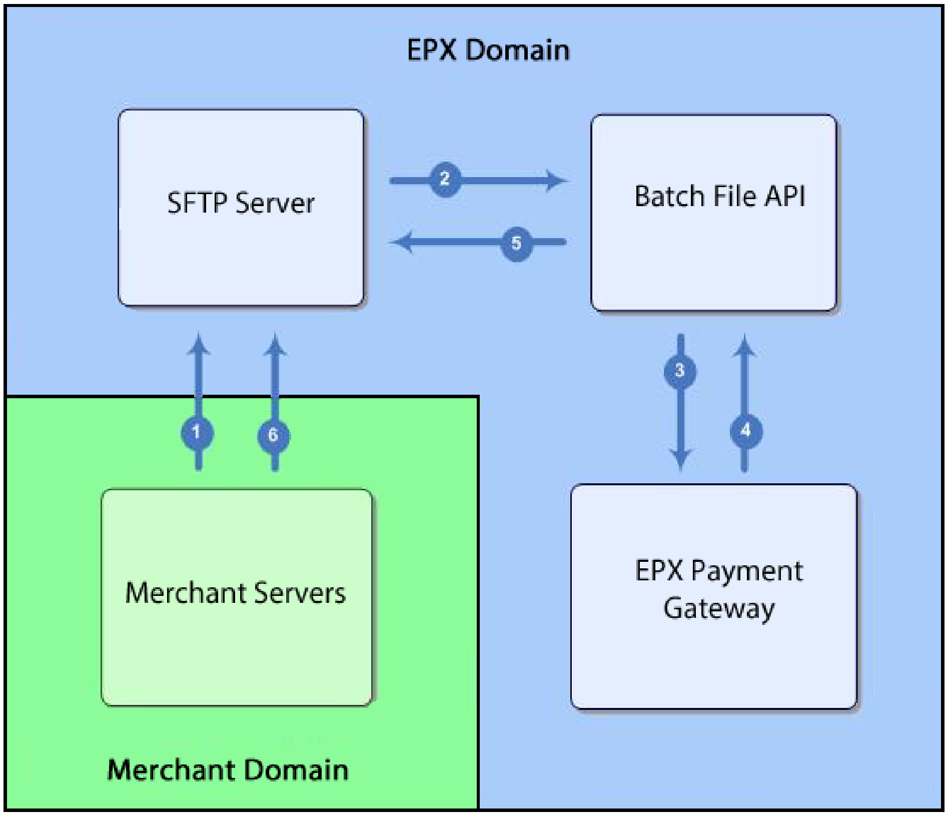
Contents

API - Batch File Overview

The EPX Batch File API is an SFTP service designed to allow the secure processing of a batch file of financial transactions from a merchant's server directly to the EPX Payment Gateway. When implemented correctly, a merchant can upload a batch file of request transactions from their servers directly to EPX for offline financial processing. This method of connecting to EPX can be used to process transactions containing full customer account information or the EPX "BRIC" token.

The implementation of the Batch File API is commonly used for merchant-driven recurring or back office transactions using a BRIC that was previously established through an alternate EPX API. This is just one of the common uses for the Batch File API that can assist in allowing the merchant to meet all of their processing needs with the EPX token technology.

Process Flow



1. The merchant uploads the XML batch file of transaction requests to their home directory of the EPX SFTP site using SFTP client on port 22 and the credentials provided by EPX.
2. The file is picked up by the Batch File API for parsing and processing.
3. The Batch File API sends each parsed transaction to the EPX Payment Gateway for processing.

4. The Batch File API receives the response for each transaction and combines those responses into an XML response file.
5. The Batch File API places the XML response file into the home directory where the request file was retrieved.
6. The merchant then reconnects to the EPX SFTP server to retrieve the XML response file for parsing.

Request Files

The Request file is formatted using XML (Extensible Markup Language). The file needs to conform to a simple layout defined by EPX. This is done using predefined XML tags.

The layout consists of three main tags, <FILE>, <BATCH>, and <DETAIL>.

Name	Description	Allowed Number of Uses
<FILE>	unique identifier	1
<BATCH>	separate different batches in the file	1+ (depends on # of 4PK, IDs)
<DETAIL>	actual transaction identifier; can contain several different sub-tags depending on the type of the request	1+ per batch

Below is a simplified example of the structure of a basic XML Batch file. Refer to the appropriate Transaction Specifications manual to view a list of mandatory and optional fields for use with the transactions in that industry.

```
<FILE>
  <BATCH>
    <DETAIL>
    </DETAIL>
    <DETAIL>
    </DETAIL>
    ...
  </BATCH>
  <BATCH>
    <DETAIL>
    </DETAIL>
    ...
  </BATCH>
  ...
</FILE>
```

Tag Attributes

The <FILE> and <BATCH> tags have attributes associated with them. For additional information about these attributes, please refer to the *Processing Data Dictionary*.

Example Request File

The following is an example Request file.

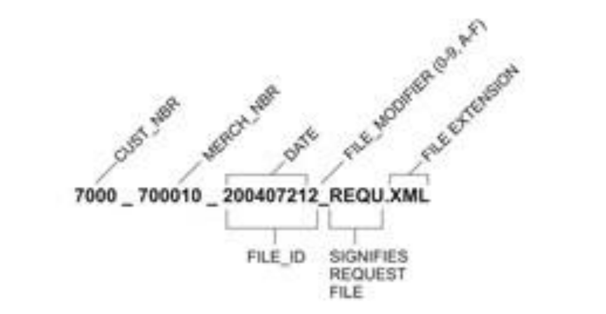
```
<FILE file_id="20070905A" file_total_amount="15.35" file_tran_count="4">
<BATCH batch_id="300" cust_nbr="1234" merch_nbr="12345" dba_nbr="1"
terminal_nbr="1" batch_total_amount="8.58" batch_tran_count="2">
<DETAIL>
<ACCOUNT_NBR>4111111111111111</ACCOUNT_NBR>
<AMOUNT>4.28</AMOUNT>
<EXP_DATE>1603</EXP_DATE>
<CARD_ENT_METH>X</CARD_ENT_METH>
<TRAN_TYPE>CCR1</TRAN_TYPE>
<TRAN_NBR>1</TRAN_NBR>
</DETAIL>
<DETAIL>
<ACCOUNT_NBR>4111111111111111</ACCOUNT_NBR>
<AMOUNT>4.30</AMOUNT>
<EXP_DATE>1603</EXP_DATE>
<CARD_ENT_METH>X</CARD_ENT_METH>
<TRAN_TYPE>CCE1</TRAN_TYPE>
<TRAN_NBR>2</TRAN_NBR>
</DETAIL>
</BATCH>
<BATCH batch_id="301" cust_nbr=“1234” merch_nbr=“12345” dba_nbr="1"
terminal_nbr="1"
batch_total_amount="6.77" batch_tran_count="2">
<DETAIL>
<ACCOUNT_NBR>4111111111111111</ACCOUNT_NBR>
<AMOUNT>5.77</AMOUNT>
<EXP_DATE>1603</EXP_DATE>
<CARD_ENT_METH>X</CARD_ENT_METH>
<TRAN_TYPE>CCE1</TRAN_TYPE>
<TRAN_NBR>1</TRAN_NBR>
</DETAIL>
<DETAIL>
<TRAN_TYPE>CKC2</TRAN_TYPE>
<ACCOUNT_NBR>12345678</ACCOUNT_NBR>
<ROUTING_NBR>123123123</ROUTING_NBR>
<TRAN_NBR>5</TRAN_NBR>
<AMOUNT>1.00</AMOUNT>
<FIRST_NAME>Joe</FIRST_NAME>
<LAST_NAME>Tester</LAST_NAME>
<ADDRESS>123 My Street</ADDRESS>
<CITY>Claymont</CITY>
<STATE>DE</STATE>
<ZIP_CODE>19703</ZIP_CODE>
</DETAIL>
</BATCH>
</FILE>
```

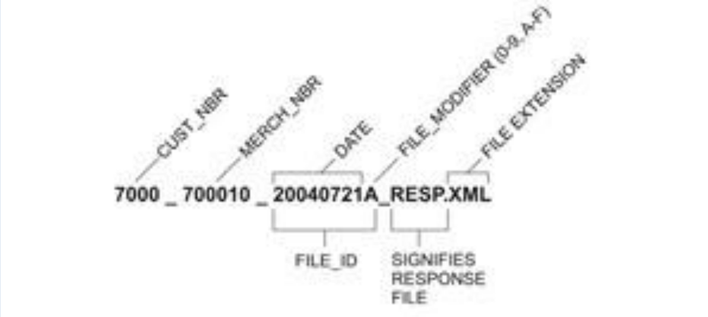
File Name Convention

The following tables contain examples of the Request and Response file name conventions with each of the values identified.

NOTE: Each value is separated by an underscore character (_).

<table>
<tr><th>Field Name</th><th>Assigned By</th><th>Format</th><th>Details</th></tr>
<tr><td>CUST_NBR</td><td>EPX</td><td>Numeric</td><td></td></tr>
<tr><td>MERCH_NBR</td><td>EPX</td><td>Numeric</td><td></td></tr>
<tr><td>DATE</td><td>Merchant</td><td>Format as <i>YYYYMMDD</i> (Part of the File ID)</td><td>FILE_ID value must match the XML tag within the file.</td></tr>
<tr><td>FILE_ MODIFIER</td><td>Merchant</td><td>0 - 9, A - Z (Part of the File ID)</td><td>FILE_MODIFIER is required for the file name to be unique and can contain up to 10 alphanumeric characters.</td></tr>
<tr><td colspan="4">Sample File Names:





</td></tr>
</table>

Transmission of Files

Before you transmit a test file to EPX, your integration specialist is required to review the format of the file. You can send your file to the Integration team at EPX and you will receive feedback on its accuracy. Once the file has been reviewed you will receive credentials for connecting to the EPX Certification environment where you can transmit your file for processing.

Send your file to EPX using the industry-standard SFTP (SSH File Transfer Protocol) to provide secure file transfer.

Use the SFTP Put command (Push) to send the file. After EPX receives the merchant's Request file, EPX forwards the authorizations to the appropriate network. Upon receiving the responses from the network, EPX generates the Response file, which you receive with the SFTP Get command (Pull).

The time between the Push of the Request file and the Pull of the Response file depends on the number of authorization requests in the Request file and the latency to the networks.

The transmission of authorization batch files is available 24/7/365. The settlement process for batch file authorizations follows the normal EPX settlement times. Refer to the *Processing Data Dictionary* for additional information.

Response File

The Response file is formatted using XML (Extensible Markup Language). The general tags are identical to those in the Request file. The <FILE> and <BATCH> tags contain the same information as received in the Request file. The <DETAIL> tag contains the Response tags that indicate the result of the authorization request.

Tag Attributes

The <DETAIL> tag contains the response that is returned from EPX. The pieces of the response are delivered in the child elements. The <DETAIL> tag has an attribute of tran_nbr, which corresponds to the <TRAN_NBR> tag that was sent as part of the detail tag in the Request file. You can use this attribute to match this response to the originating authorization request. Below is a sample structure of a credit card response and an ACH response. For additional information, refer to the *Processing Data Dictionary*.

Fields included in the Response are subject to change, with additional features regularly being added. Processing a transaction in the certification environment will give you an accurate example of fields that will be included in the production EPX Response.

Example Response File

Below is an example Response file.

```
<FILE file_id = "20070905A" >
<BATCH batch_id = "300" cust_nbr = “1234” merch_nbr = “12345” dba_nbr = "1"
terminal_nbr = "1" >
<DETAIL tran_nbr="1">
<AUTH_GUID>0V7006LLX0T00W33LUL</AUTH_GUID>
<AUTH_RESP>00</AUTH_RESP>
<AUTH_CODE>000458</AUTH_CODE>
<AUTH_AVS> </AUTH_AVS>
<AUTH_CVV2> </AUTH_CVV2>
<AUTH_RESP_TEXT>APPROVAL </AUTH_RESP_TEXT>
<AUTH_SETTLE_DT></AUTH_SETTLE_DT>
<AUTH_LEDGER_BAL></AUTH_LEDGER_BAL>
<AUTH_AVAILABLE_BAL></AUTH_AVAILABLE_BAL>
<AUTH_CARD_TYPE>V</AUTH_CARD_TYPE>
<TRAN_TYPE>CCR1</TRAN_TYPE>
<AUTH_TRAN_DATE_GMT>09/05/2007 04:30:50 PM</AUTH_TRAN_DATE_GMT>
</DETAIL>
<DETAIL tran_nbr ="2" >
<AUTH_GUID>0V7006LLX3R00HTNLUM</AUTH_GUID>
<AUTH_RESP>00</AUTH_RESP>
<AUTH_CODE>000460</AUTH_CODE>
<AUTH_AVS> </AUTH_AVS>
<AUTH_CVV2> </AUTH_CVV2>
```

```
<AUTH_RESP_TEXT>APPROVAL </AUTH_RESP_TEXT>
<AUTH_SETTLE_DT></AUTH_SETTLE_DT>
<AUTH_LEDGER_BAL></AUTH_LEDGER_BAL>
<AUTH_AVAILABLE_BAL></AUTH_AVAILABLE_BAL>
<AUTH_CARD_TYPE>V</AUTH_CARD_TYPE>
<TRAN_TYPE>CCE1</TRAN_TYPE>
<AUTH_TRAN_DATE_GMT>09/05/2007 04:30:50 PM</AUTH_TRAN_DATE_GMT>
</DETAIL>
</BATCH>
<BATCH batch_id = "301" cust_nbr = “1234” merch_nbr = “12345” dba_nbr = "1"
terminal_nbr = "1" >
<DETAIL tran_nbr ="1" >
<AUTH_GUID>0V7006LLX59001FZLUN</AUTH_GUID>
<AUTH_RESP>00</AUTH_RESP>
<AUTH_CODE>000459</AUTH_CODE>
<AUTH_AVS> </AUTH_AVS>
<AUTH_CVV2> </AUTH_CVV2>
```