Account Updater

Developer Guide

BANK OF AMERICA

Restricted Rights Legends

For Government or defense agencies: Use, duplication, or disclosure by the Government or defense agencies is subject to restrictions as set forth the Rights in Technical Data and Computer Software clause at DFARS 252.227-7013 and in similar clauses in the FAR and NASA FAR Supplement.

For civilian agencies: Use, reproduction, or disclosure is subject to restrictions set forth in subparagraphs (a) through (d) of the Commercial Computer Software Restricted Rights clause at 52.227-19 and the limitations set forth in Cybersource Corporation's standard commercial agreement for this software. Unpublished rights reserved under the copyright laws of the United States.

Trademarks

Authorize.Net, eCheck.Net, and The Power of Payment are registered trademarks of Cybersource Corporation. Cybersource, Cybersource Payment Manager, Cybersource Risk Manager, Cybersource Decision Manager, and Cybersource Connect are trademarks and/or service marks of Cybersource Corporation. Visa, Visa International, Cybersource, the Visa logo, the Cybersource logo, and 3-D Secure are the registered trademarks of Visa International in the United States and other countries. All other trademarks, service marks, registered marks, or registered service marks are the property of their respective owners.

Confidentiality Notice

This document is furnished to you solely in your capacity as a client of Cybersource and as a participant in the Visa payments system.

By accepting this document, you acknowledge that the information contained herein (the Information) is confidential and subject to the confidentiality restrictions contained in Visa's operating regulations and/or other confidentiality agreements, which limit your use of the Information. You agree to keep the Information confidential and not to use the Information for any purpose other than its intended purpose and in your capacity as a customer of Cybersource or as a participant in the Visa payments system. The Information may only be disseminated within your organization on a need-to-know basis to enable your participation in the Visa payments system. Please be advised that the Information may constitute material non-public information under U.S. federal securities laws and that purchasing or selling securities of Visa Inc. while being aware of material non-public information would constitute a violation of applicable U.S. federal securities laws.

Additional Notices

This document may contain depictions of a product currently in the process of deployment, and should be understood as a representation of the potential features of the fully-deployed product. The final version of this product may not contain all of the features described in this document.

Any timelines and roadmaps contained in this document are subject to change at Cybersource's sole discretion.

The data on this document are used for illustration only and do not reflect actual Cybersource data.

Case studies, comparisons, statistics, research and recommendations are provided AS IS and intended for informational purposes only and should not be relied upon for operational, marketing, legal, technical, tax, financial or other advice. The Information contained herein is not intended as investment or legal advice, and readers are encouraged to seek the advice of a competent professional where such advice is required.

Version: 22.01

Contents

Recent Revisions to This Document

Release	Changes
22.01	Terminology updates.
21.04	Added information for Legacy CSV Reports (on page 28).
21.03	Updated product names throughout the document.
21.02	Updated URLs throughout the document. Added error codes. See Error Codes (on page 27).
21.01	Added information about tokenization. See Tokenization (on page 6). Added information about token updates. See Token Updates (on page 9). Added American Express test cards. See American Express Test Card Numbers (on page 42). Added REST API fields. See API Fields (on page 46).
20.01	Initial release.

Introduction

Account Updater helps you to keep stored card data up to date so that you can improve authorization success rates by cutting down on declined payments related to lost, stolen, or expired cards. Account Updater updates card data stored on your servers. The updates include expiration dates, credit card numbers, and brands.

Account Updater provides a single interface to access updates from the Visa Account Updater and Mastercard Automatic Billing Updater services. If you are using Token Management, Bank of America also provides updates to you from the American Express Cardrefresher service.

Integration options are available depending on whether you are:

- Using Token Management or Recurring Billing for tokenization. See Tokenization (on page 6).
- Storing PANs your system. See Storing PANs (on page 7).

After you review this guide, use the Developer Center to access sample code (in addition to the examples in this document) and details about required fields for your API requests. The Developer Center was developed by our vendor and uses some terminology and branding that are slightly different from the other materials on the Merchant Help Center. Later in this document, we provide explanations where needed so that you can navigate the Developer Center more easily.

Tokenization

If you are already using Token Management or Recurring Billing, Account Updater is simple to integrate. You will benefit from updates from Visa, Mastercard, and American Express.

All Bank of America Merchant Services clients have the option to create and use customer tokens and billing tokens. These are tokenized, secure instruments for storing customer and card data. If you use the token Harvest or the token batch update method for Account Updater, the service will update your customer tokens and billing tokens.

> ⓘ **Important:** If you are using Bank of America Token Management, review the updates that you receive from the card networks to determine if updates to your systems are required to optimize your processes. For example, if you present the last four digits of the card, you should make any necessary updates to your system(s) to ensure the last four digits of a new replacement card (NAN) are presented with any online screens presented to your customer.

To use Account Updater, you must generate the REST API keys in the key management section of Business Advantage 360, under Merchant Services. Business Advantage 360 is our Small Business Online Banking platform, where you can access your Merchant Services account.

Token Harvest Updates

You can configure Account Updater to automatically update all of your tokens with the latest credit card data. For Visa and Mastercard, update reports are generated monthly. For American Express, update reports are generated daily. See Token Harvest (on page 9).

Token Batch Updates

Using the Account Updater REST API, you can add the specific tokens (also known as subscriptions) that you wish to batch update. For Visa and Mastercard, these batches produce one-off update reports.

For tokens containing American Express cards, card numbers are registered and enrolled for automatic updates for which reports are generated daily. Tokens are removed automatically when deleted or updated to a different card type. See Token Batch Updates (on page 10).

Storing PANs

When you directly manage customer card data, you create a file containing PANs that Account Updater updates. Create a request file containing new PANs and POST it to the Account Updater URL. Download the response file from within your Merchant Services account in Business Advantage 360 or a client application. See PAN Updates (on page 29).

Enabling Account Updater

You must sign up for Account Updater directly with your merchant consultant. Then Bank of America will submit enrollment form on your behalf to Mastercard and Visa. The enrollment process can take up to 12-14 business days.

Terms of Use

By using the Account Updater service, you agree to comply with the Visa U.S.A. Operating Regulations, Visa Account Updater Terms of Use, Mastercard rules and regulations, American Express rules and regulations, and all other applicable rules and regulations issued by any card association.

In addition, you must:

- Request an update for every participating Visa account in your customer database at least:
  - Once every 180 calendar days if you bill daily, weekly, monthly, quarterly, or biannually.
  - Once every 365 calendar days if you bill annually.
- Submit inquiries only for those accounts with which you have an ongoing customer relationship.
- Update your customer account database within 5 business days of receiving an update.
- Ensure that all update information you receive is properly, completely, and accurately incorporated into your data store for use in future transactions.
- Correct erroneous account information within 5 business days of receipt of error notifications.

You may not:

- Request updates on accounts that have returned a response of Contact Card Holder (CCH). You must review your response file for CCH responses and take appropriate action such as removing the customer record from your billing cycle until you have contacted the cardholder.
- Submit update inquiries on behalf of any other entity.

Token Updates

You can arrange for Account Updater to harvest and update all of your tokens on an agreed-upon date; this is called the *token harvest update* method. You must retrieve a monthly report for Visa and Mastercard updates and a daily update report for American Express Cardrefresher.

Account Updater requires card number and expiration dates, so the token harvest option is available only when you use the *customer* or *billing* tokens.

The Account Updater REST API enables you to selectively send a POST request for a batch of tokens (subscription IDs) for American Express cards to be enrolled, processed, and updated. This is called the *token batch update* method.

Both options use the standard REST API authentication methods:

- JSON Web Token (JWT)
- HTTP signature

For information about REST API authentication methods, see the *Bank of America Gateway Integration Guide* on the Merchant Help Center, which includes a link to the Developer Center in the Help Center.

Token Harvest

On an agreed-upon monthly date, Account Updater submits your tokenized cards to Visa and Mastercard for updates. American Express tokenized cards are automatically enabled for Cardrefresher daily, and deleted or updated tokens are de-enrolled automatically.

You must retrieve American Express reports daily and/or Visa and Mastercard reports monthly. For more information, see the Retrieving Update Reports (on page 14).

It is best practice to request updates for your tokens 3 to 5 days before your billing cycle begins. You can choose any calendar day, from the 1st through the 28th.

Token Batch Updates

Batches Resource

To access endpoints, use an HTTPS POST request with a valid JSON payload:

- Test: `https://apitest.merchant-services.bankofamerica.com/accountupdater/v1/batches`
- Production: `https://api.merchant-services.bankofamerica.com/accountupdater/v1/batches`

Submitting Visa and Mastercard One-Off Updates

You can submit a specific set of tokens or all tokens for a one-off update to Visa and Mastercard. Your update report is generated in 24 to 48 hours. A successful response to the batch creation returns a batch ID. You can check the status of the batch, which returns the URI of the batch update report when available.

To perform a one-off update, set the **type** field to `oneOff` in order.

Registering Tokens for American Express Daily Updates

Register tokens containing American Express cards for daily updates. Account Updater receives updates from American Express daily, applies them to your tokens, and produces a daily report that is available to you through the REST API.

To indicate that the batch contains tokens to be enrolled with American Express Cardrefresher, set the **type** to `amexRegistration`.

American Express registration registers tokens for daily updates. You will receive American Express updates as they occur in the daily report. You will not see a batch response file for American Express tokens in the manner that you receive batch responses for Visa and Mastercard batch submissions.

Batch Creation Request

Token Management supports different types of tokens. In the Developer Center and in the code examples you'll see in this document, you will see different terminology than how these tokens are identified in the Merchant Help Center. These terms are interchangeable.

Customer Token

Represents data about the merchant's customer including email address, customer ID, shipping address (stored in a token), and other related fields

Payment Instrument

Represents the complete billing details for the payment type, including cardholder name, expiration date, and billing address.

Instrument Identifier

Represents the tokenized primary account number (PAN) for card payments, as well as the associated COF Network Token, or U.S. or Canadian bank account number and routing number.

Merchant Help Center terminology:

- Customer Token
- Billing Token
- PAN Token

API and code example terminology:

- Customer Token
- Payment Instrument
- Instrument Identifier

Customer tokens and *payment instrument* tokens store the expiration date in addition to the PAN. *Instrument identifier* tokens store only the PAN.

Each batch request should contain only one token type: *customer*, *payment instrument*, or *instrument identifier*.

Account Updater requires the existing PAN and expiration date. If you are using *instrument identifier* tokens, you must also to specify the expiration date.

Example: Creating a Batch of Two Customer or Payment Instrument Tokens

```
{
  "type": "oneOff",
  "included": {
    "tokens": [
      {
        "id": "3FA02EB4E49B65FDA194B38994B1F3F3"
      },
      {
        "id": "D1944BD9A7F9052BE431A276EB492C39"
      }
    ]
  },
  "merchantReference": "Merchant reference",
  "notificationEmail": "email@example.com"
}
```

Example: Creating a Batch of Two Instrument Identifier Tokens

```
{
  "type": "amexRegistration",
  "included": {
    "tokens": [
      {
        "id": "6725011400587861",
          "expirationMonth": "12",
          "expirationYear": "2021"
      },
      {
        "id": "6725011400587862",
          "expirationMonth": "12",
          "expirationYear": "2021"
      }
    ]
  },
  "merchantReference": "Merchant reference",
  "notificationEmail": "email@example.com"
}
```

Batch Creation Response Examples

Example: HTTP 202 – Successful batch creation

```
{
  "_links": {
    "self": {
      "href":
 "https://api.merchant-services.bankofamerica.com/accountupdater/v1/batches"
    },
      "status": {
      "href":
 "https://api.merchant-services.bankofamerica.com/accountupdater/v1/batches/152699
96945240002139594385/status"
    }
  },
  batchId": "15269996945240002139594385",
  batchItemCount": 2
}
```

Example: HTTP 401 – Not authorized to access resource

```
{
  "_links": {
    "self": {
      "href":
 "https://api.merchant-services.bankofamerica.com/accountupdater/v1/batches"
    }
  },
  "code": "FORBIDDEN_RESPONSE",
  "correlationId": "c7b74452a7314f9ca28197d1084447a5",
  "detail": "You are not authorized to access this resource",
  "fields": null,
  "localizationKey": "cybsapi.forbidden.response",
  "message": "Unauthorized Access"
}
```

Action: Verify that the credentials that you are using are correct for the environment you are accessing. Ensure that your credentials have not expired and that your authentication process is correct.

Example: HTTP 422 – Failure to process request

```
{
  "_links": {
    "self": {
      "href":
 "https://api.merchant-services.bankofamerica.com/accountupdater/v1/batches"
    }
  },
  "code": "VALIDATION_ERROR",
  "correlationId": "c7b74452a7314f9ca28197d1084447a5",
  "detail": "One or more fields failed validation",
  "fields": [
    {
      "path": "notificationEmail",
      "message": "Email address provided should not be 'null'",
      "localizationKey": "cybsapi.ondemand.batch.email.null"
    }
  ],
  "localizationKey": "cybsapi.validation.error",
  "message": "Field validation error"
}
```

Action: Examine the message to learn what failed validation. Verify that the structure of your JSON format is correct.

Retrieving Update Reports

The update reports contain details of updates that have been applied to the tokens in your batch, and include a masked version of new card numbers and/or expiration dates.

To retrieve the batch, obtain the batch ID. The process for retrieving the batch depends on how the batch was created.

Visa and Mastercard One-Off Updates

To retrieve one-off updates, verify the batch status URL that was returned in the one-off batch creation. See Submitting Visa and Mastercard One-Off Updates (on page 10).

To get the status of the batch, send an authenticated GET request, including the header `ACCEPT=application/json`, to one of the following resources:

- Test: `https://apitest.merchant-services.bankofamerica.com/accountupdater/v1/batches/{batchId}/status`
- Production: `https://api.merchant-services.bankofamerica.com/accountupdater/v1/batches/{batchId}/status`

One-Off Batch Update Response

Batch processing by Visa and Mastercard can take up to 48 hours; therefore, reports are not available immediately. A successful response returns the status of the batch and additional information relating to the batch as it becomes available.

These batch statuses are possible:

Status Responses

Status	Description
`Received`	The batch was received and is being checked for errors.
`Processing`	The batch was sent to the card association(s) to be updated.
`Updating`	Account Updater received a response from the card association(s) and is updating the tokens.
`Completed`	Updates have been applied to the tokens. The batch report URL is now available.
`Failed`	Review specific error message.

Not all data is available immediately. As the batch status progresses from `Received` through `Processing` and `Updating` to `Completed`, additional data becomes available in the batch status. Check the status after submitting the batch to catch early errors that might result in a `Failed` status or incorrect `acceptedRecords` or `rejectedRecords` counts. The URL of the batch report appears when the status is `Completed`.

Example: HTTP 200 – Successful response

```
{
  "_links": {
    "self": {
      "href":
 "https://api.merchant-services.bankofamerica.com/accountupdater/v1/batches/152699
96945240002139594385/status"
    },
    "report": [
      {
      "href":
 "https://api.merchant-services.bankofamerica.com/accountupdater/v1/batches/152699
96945240002139594385/status"
      }
    ]
    "batchCaEndpoints": "VISA,MASTERCARD",
    "batchCreatedDate": "2018-05-22T14.38.57Z",
    "batchId": "15269996945240002139594385",
    "batchSource": "TOKEN_API",
    "billing": {
      "nan": "0,",
      "ned": "9,",
      "acl": "5,",
      "cch": 0
    },
    "description": "Batch processing complete. Report URL now available.",
    "merchantReference": "Merchant reference",
    "status": "COMPLETED",
    "totals": {
      "acceptedRecords": "8,",
      "rejectedRecords": "7,",
      "updatedRecords": "8,",
      "caResponses": "14,",
      "caResponsesOmitted": 6
    }
}
```

American Express Daily and Token Harvest Update Reports

American Express update reports are generated daily, so the batch ID is not known in advance.

Similarly, token harvest updates are scheduled by the Account Updater service on a date that you agree upon with your merchant consultant..

For daily and token harvest update reports, the first step is to retrieve the batch ID itself by sending an authenticated GET request, including the header `ACCEPT=application/json`, to one of the following resources:

- Test: `https://apitest.merchant-services.bankofamerica.com/accountupdater/v1/batches`
- Production: `https://api.merchant-services.bankofamerica.com/accountupdater/v1/batches`

The response is an array of batches. Paging is supported with offset and limit query parameters. For example, to return the second page of results with 50 per page, send `/v1/batches?offset=1&limit=50`.

To filter by date, add the **fromDate** and **toDate** fields as a query string using the UTC date format yyyymmddThhmmssZ. Example: `v1/batches?fromDate=20200315T000000Z&toDate=20200415T000000Z`.

Example: HTTP 200 – Successful Response

```
{
  "_links": [
    {
      "rel": "self",
      "href":
 "https://apitest.merchant-services.bankofamerica.com/accountupdater/v1/batches?of
fset=0&limit=1"
    },
    {
      "rel": "first",
      "href":
 "https://apitest.merchant-services.bankofamerica.com/accountupdater/v1/batches?of
fset=0&limit=1"
    },
    {
      "rel": "next",
      "href":
 "https://apitest.merchant-services.bankofamerica.com/accountupdater/v1/batches?of
fset=1&limit=1"
    },
    {
```

Example: HTTP 200 – Successful Response (*continued*)

```
      "rel": "last",
      "href":
 "https://apitest.merchant-services.bankofamerica.com/accountupdater/v1/batches?of
fset=114&limit=1"
    }
  ],
  "object": "collection",
  "offset": 0,
  "limit": 3,
  "count": 1,
  "total": 3,
  "_embedded": {
    "batches": [
      {
        "_links": {
        "reports": [
          {
            "href":
 "https://apitest.merchant-services.bankofamerica.com/accountupdater/v1/batches/15
416031479410002099212314/report"
            }
          ]
        },
        "batchId": "15416031479410002099212314",
        "batchCreatedDate": "2018-11-07T07:05:48Z",
        "batchModifiedDate": "2018-11-07T07:05:50Z",
        "batchSource": "SCHEDULER",
        "tokenSource": "Token Management",
        "merchantReference": "Merchant Name",
        "batchCaEndpoints": [
          "VISA",
          "MASTERCARD"
        ],
        "status": "COMPLETE",
        "totals": {
          "acceptedRecords": 1,
          "rejectedRecords": 0,
        "updatedRecords": 1,
        "caResponses": 1,
        "caResponsesOmitted": 0
        }
      },
```

Example: HTTP 200 – Successful Response (*continued*)

```
    {
      "_links": {
        "reports": [
          {
            "href":
 "https://apitest.merchant-services.bankofamerica.com/accountupdater/v1/batches/15
416025010730001655343827/report"
              }
            ]
          },
          "batchId": "15416025010730001655343827",
          "batchCreatedDate": "2018-11-07T06:55:01Z",
          "batchModifiedDate": "2018-11-07T06:56:52Z",
          "batchSource": "AMEX_REGISTRY_API",
          "tokenSource": "Token Management",
          "batchCaEndpoints": [
          "AMEX"
          ],
          "status": "COMPLETE"
        },
      {
        "_links": {
          "reports": [
            {
              "href":
 "https://apitest.merchant-services.bankofamerica.com/accountupdater/v1/batches/15
416025010730001655343827/report"
            }
          ]
        },
        "batchId": "15402221273070001683984545",
        "batchCreatedDate": "2018-10-22T08:28:47Z",
        "batchModifiedDate": "2018-10-22T08:29:19Z",
        "batchSource": "AMEX_MAINTENANCE",
        "tokenSource": "Token Management",
        "batchCaEndpoints": [
        "AMEX"
        ],
        "status": "COMPLETE",
        "totals": {
        "acceptedRecords": 0
        }
      }
    ]
  }
}
```

Batches are identified by the **batchCreatedDate** and the **batchSource** field values. Possible **batchSource** values:

Batch Source Values

batchSource Values	Description
`AMEX_REGISTRY_API`	Batch for American Express token registration. American Express generates a report only when the registration batch contains errors.
`AMEX_MAINTENANCE`	Daily updates for tokens enrolled in the American Express Cardrefresher service.
`TOKEN_API`	Updates relating to a one-off request to Visa or Mastercard.
`SCHEDULER`	Updates relating to a monthly harvest of all tokens.

After you submit a batch for American Express token registration, you can access the batch status through the authenticated GET request using the URL returned in the submission. A successful response returns the status of the batch. See Registering Tokens for American Express Daily Updates (on page 10).

Example: American Express Registry Status Response

```
{
  "_links": {
    "self": {
      "href":
 "https://apitest.merchant-services.bankofamerica.com/accountupdater/v1/batches/15
816023535620001646854894/status"
      },
      "report": [
{
        "href":
 "https://apitest.merchant-services.bankofamerica.com/accountupdater/v1/batches/15
816023535620001646854894/report"
      }
    ]
  },
  "batchCaEndpoints": "AMEX",
  "batchCreatedDate": "2020-02-13T13.59.13Z",
  "batchId": "15816023535620001646854894",
  "batchSource": "AMEX_REGISTRY_API",
  "description": "Updates have been applied to your tokens. A batch report is
 available.",
  "merchantReference": "Merchant Name",
  "status": "COMPLETE",
  "totals":
  {
    "acceptedRecords": 999,
    "rejectedRecords": 123,
    "updatedRecords": 0,
    "caResponses": 0,
    "caResponsesOmitted": 0
  }
}
```

Retrieving a Batch with a Batch ID

To access an individual batch report, send an authenticated GET request, including the header `ACCEPT=application/json`, using the URL returned in the batch status or batches resource described in American Express Daily Harvest Update Reports (on page 17).

Example: AMEX_REGISTRY_API Batch Method HTTP 200 – Successful response

```
{
  "version": "1.0",
  "reportCreatedDate": "2018-11-07T15:33:11Z",
  "batchId": "15416047164330001593314231",
  "batchSource": "AMEX_REGISTRY_API",
  "batchCaEndpoints": "AMEX",
  "batchCreatedDate": "2018-11-07T15:31:56Z",
  "merchantReference": "Merchant Name",
  "totals": {
    "acceptedRecords": 0,
    "rejectedRecords": 3
  },
  "records": [
    {
      "sourceRecord": {
        "token": "12345678901234567890",
        "cardExpiryMonth": "01",
        "cardExpiryYear": "2021"
  },
  "responseRecord": {
    "response": "DEC",
    "reason": "852"
  }
},
{
  "sourceRecord": {
    "token": "456",
    "cardExpiryMonth": "01",
    "cardExpiryYear": "2021"
  },
  "responseRecord": {
    "response": "DEC",
    "reason": "851"
  }
},
```

Example: AMEX_REGISTRY_API Batch Method HTTP 200 – Successful response (*continued*)

```
{
  "sourceRecord": {
     "token": "789",
     "cardExpiryMonth": "01",
     "cardExpiryYear": "2021"
  },
  "responseRecord": {
     "response": "DEC",
     "reason": "851"
      }
    }
  ]
}
```

American Express Daily Updates

Card numbers in Token Management are represented by *instrument identifier* tokens. A card number and its associated *instrument identifier* token is set to a CLOSED status when:

- The card network sends a direct account closed notification (response code ACL).
- A new card number is issued to replace a cancelled card (response code NAN).

Account Updater updates *customer* and *payment instrument* tokens only when you specify them in the request. When you specify a *customer* token for update or harvest, only the customer's default *payment instrument* token is updated. When you do not specify the *customer* and *payment instrument* tokens, they can become associated with a closed *instrument identifier* token in the update batch or harvest. These results are detailed in the `additionalUpdates` section of the update report. To update *customer* tokens and *payment instrument* tokens, include them in a subsequent Account Updater batch API request, or send a direct call to the REST API.

Example: AMEX_MAINTENANCE Batch Method

```
{
  "version": "1.0",
  "reportCreatedDate": "2020-01-23T11:16:13Z",
  "batchId": "15797780137010000506182090",
  "batchSource": "AMEX_MAINTENANCE",
  "batchCaEndpoints": "AMEX",
  "batchCreatedDate": "2020-01-23T11:13:33Z",
  "totals": {
    "updatedRecords": 3,
    "rejectedRecords": 0,
```

Example: AMEX_MAINTENANCE Batch Method (*continued*)

```
   "caResponses": 3,
   "caResponsesOmitted": 0
 },
 "billing": {
   "nan": 1,
   "ned": 1,
   "acl": 1,
   "cch": 0
 },
 "records": [
   {
     "id": "562239661",
     "sourceRecord": {
       "token": "9CCD3AE24DD9E254E0533F36CF0A356E",
       "cardNumber": "371449XXXXX2009",
       "cardExpiryMonth": "02",
       "cardExpiryYear": "2021",
       "cardType": "003",
       "customerId": "9CCD3AE24DD9E254E0533F36CF0A356E",
       "paymentInstrumentId": "9CCD3AE24DD8E254E0533F36CF0A356E",
       "instrumentIdentifierId": "6725011400587863"
     },
     "responseRecord": {
       "response": "NAN",
       "reason": "800",
       "token": "9CCD3AE24DD9E254E0533F36CF0A356E",
       "cardNumber": "371449XXXXX0102",
       "cardType": "003",
       "instrumentIdentifierId": "6725011400587864",
       "instrumentIdentifierCreated": "true",
       "cardExpiryMonth": "07",
       "cardExpiryYear": "2021",
       "additionalUpdates": [
         {
           "customerId": "8CCD3AE24DD8E254E0533F36CF0A355E",
           "paymentInstrumentId": "9CCD3AE24DD8E254E0533F36CF0A356D",
           "creator": "aura_regress_tms_report",
           "state": "CLOSED",
           "message": "This Payment Instrument contains the source card number,
which is now closed. If required, you can update manually or through the AU REST
API."
         }
       ]
     }
   }
```

Example: AMEX_MAINTENANCE Batch Method (*continued*)

```
  {
    "id": "562239711",
    "sourceRecord": {
      "token": "9CCD3AE24DF7E254E0533F36CF0A356E",
      "cardNumber": "371449XXXXX1100",
      "cardExpiryMonth": "02",
      "cardExpiryYear": "2021",
      "cardType": "003",
      "customerId": "9CCD3AE24DF7E254E0533F36CF0A356E",
      "paymentInstrumentId": "9CCD3AE24DF6E254E0533F36CF0A356E",
      "instrumentIdentifierId": "6725011400587865"
    },
    "responseRecord": {
      "response": "NED",
      "reason": "800",
      "cardExpiryMonth": "12",
      "cardExpiryYear": "2021"
      }
    },
    {
    "id": "562239751",
    "sourceRecord": {
      "token": "9CCD3AE24E0FE254E0533F36CF0A356E",
      "cardNumber": "371449XXXXX1226",
      "cardExpiryMonth": "02",
      "cardExpiryYear": "2021",
      "cardType": "003",
      "customerId": "9CCD3AE24E0FE254E0533F36CF0A356E",
      "paymentInstrumentId": "9CCD3AE24E0EE254E0533F36CF0A356E",
      "instrumentIdentifierId": "6725011400587866"
    },
    "responseRecord": {
      "response": "ACL",
      "reason": "800",
      "additionalUpdates": [
        {
          "customerId": "7CCD3AE24DD8E254E0533F36CF0A356A",
          "paymentInstrumentId": "9CCD3AE24E0EE254E0533F36CF0A356D",
          "creator": "aura_regress_tms_report",
          "state": "CLOSED",
          "message": "This Payment Instrument contains the source card number,
 which is now closed. If required, you can update manually or through the AU REST
 API."
          }
        ]
      }
    }
  ]
}
```

Example: TOKEN_API Batch Method and Scheduler

```
{
  version": "1.0
  "reportCreatedDate": "2018-11-01T14:43:36Z",
  "batchId": "1541083347340000123332450",
  "batchSource": "SCHEDULER",
  "batchCaEndpoints": "VISA,MASTERCARD",
  "batchCreatedDate": "2018-11-01T14:42:27Z",
  "merchantReference": "Merchant Name",
  "totals": {
    "acceptedRecords": 2,
    "caResponses": 3,
    "rejectedRecords": 0,
    "updatedRecords": 2,
    "caResponsesOmitted": 1
    },
    "billing": {
      "nan": 1,
      "ned": 0,
      "acl": 1,
      "cch": 0
    },
    "records": [
      {
        "id": "4451434614",
        "sourceRecord": {
          "token": "4682345889876532701018",
          "cardNumber": "511111XXXXXX3604",
          "cardExpiryMonth": "09",
          "cardExpiryYear": "2021",
          "cardType": "002"
        },
        "responseRecord": {
          "response": "ACL",
          "reason": "800"
        }
      },
    {
      "id": "784311",
      "sourceRecord": {
      "token": "7020000000014008934",
      "cardNumber": "371000XXXXXX8115",
      "cardExpiryMonth": "01",
      "cardExpiryYear": "2021",
      "cardType": "003",
      "instrumentIdentifierId": "7020000000014008115"
      },
```

Example: TOKEN_API Batch Method and Scheduler (*continued*)

```
      "responseRecord": {
      "response": "NAN",
      "reason": "800",
      "token": "7020000000012513358",
      "cardNumber": "401000XXXXXX2753",
      "cardType": "001",
      "instrumentIdentifierId": "7020000012512753",
      "instrumentIdentifierCreated": "true",
      "cardExpiryMonth": "08",
      "cardExpiryYear": "2021"
      }
    }
  ]
}
```

Example: Batch Retrieval Error

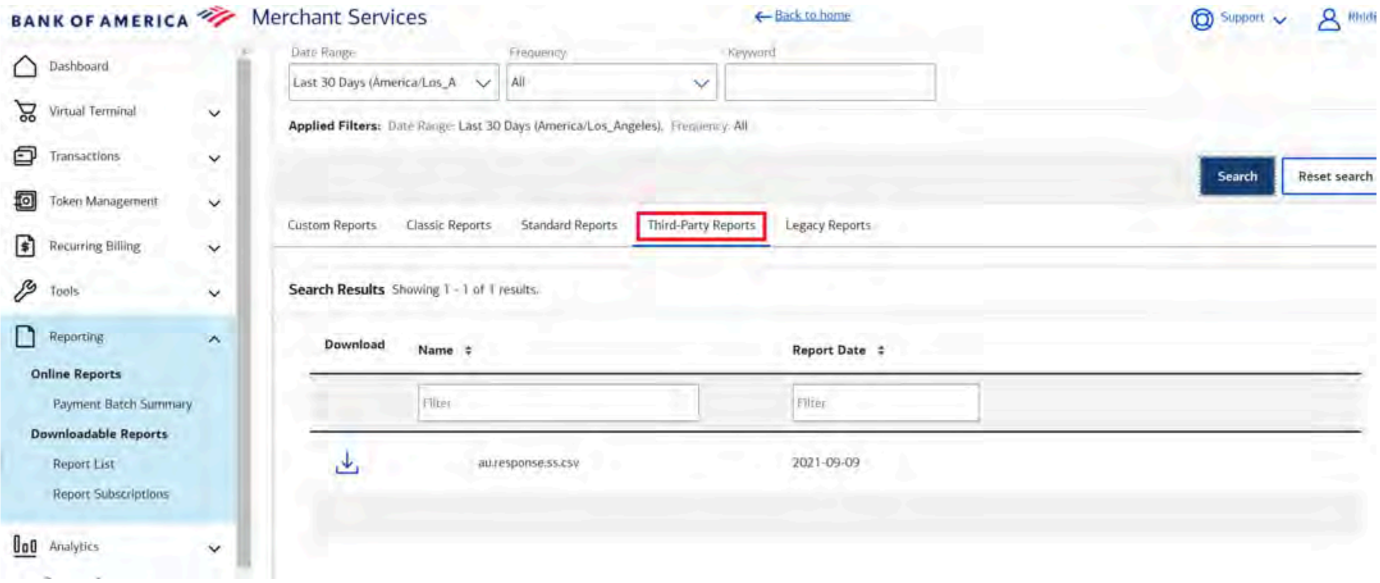
```
{
  "_links": {
    "self": {
      "href":
 "https://api.merchant-services.bankofamerica.com/accountupdater/v1/batches/
15410833473400333245O/report"
    }
  },
  "code": "FORBIDDEN_RESPONSE",
  "correlationId": "0386623ab0eb47dfae61d273032f8202",
  "detail": "You are not authorized to access this resource",
  "localizationKey": "cybsapi.forbidden.response",
  "message": "Unauthorized Access"
```

Error Codes

Response Code	Reason Code	Description	Billable
ERR	820	Card number rejected by external token provider.	No
ERR	821	External token provider service issue.	No

Legacy CSV Reports

While Bank of America recommends that your systems consume the JSON batch reports described in Token Batch Updates (on page 10) to respond to updates, there is a legacy Account Updater report for harvested Visa and Mastercard tokens available under the Third-Party Reports tab in the Report List section of your Merchant Services account.



Date Range: Last 30 Days (America/Los_A

Frequency: All

Keyword:

Applied Filters: Date Range: Last 30 Days (America/Los_Angeles), Frequency: All

Search Results Showing 1 - 1 of 1 results.

Download	Name	Report Date
	au.response.ss.csv	2021-09-09

The legacy Account Updater report provides the following data:

- Header or data indicator
- Account Updater transaction ID
- Token ID
- New card number
- Response code
- Reason code
- Old card number
- Old expiry month
- Old expiry year
- New expiry month
- New expiry year

PAN Updates

Important: You must enroll in Account Updater and comply with the Terms of Use.

For PAN updates, Account Updater files are processed once per day. You can expect your response file to be available 24 to 48 hours after you submit your request file. Send your Account Updater request file 3 to 5 days before your billing cycle starts to ensure that your file completes processing and that you have enough time to update your data store.

Responses from Visa and Mastercard are consolidated and returned in an encrypted response file. See Response File Records (on page 39).

Encrypting the PAN Data

The Account Updater Batch Upload Web Application (AU-BUWA) enables you to upload encrypted batches of PAN data to Account Updater. The AU-BUWA endpoint accepts PGP-encrypted CSV data and is secured with TLS 1.2.

PGP encryption follows the OpenPGP standard defined in RFC 4880 using a combination of symmetric and public-key encryption. Each upload uses a single-use symmetric key to encrypt the batch PAN CSV data. You send the encrypted data and symmetric session key to Account Updater, which is protected during the transmission with asymmetric encryption using your public key. Only the AU-BUWA file encryption PGP private key can decrypt the session key and use it to symmetrically decrypt the data.

Use the Account Updater issued AU-BUWA public PGP key to encrypt the data then sign each file using your merchant-specific private key.

Up to 10 MB of unencrypted CSV PAN data can be compressed, PGP-encrypted, and uploaded to the AU-BUWA encrypted PAN endpoint. AU-BUWA rejects data files that are larger than 10 MB before they are encrypted.

PGP Public/Private Key Pair Roles

To send encrypted PANs you must have two PGP public/private key pairs to encrypt the PAN CSV data file for the secure transfer and to verify the integrity and origin of the data file.

AU-BUWA PGP Key Pair

This key pair is used for encryption and decryption.

- **Public key**: use this public key to encrypt PAN CSV data prior to transferring it to the AU-BUWA encrypted PANs endpoint.
  - MD5 for the test public key: 52e654221f12a47c38f55e45c446fea0
  - MD5 for the live public key: af99bb76b328c2103c9044d29ae1f18
- **Private key**: AU-BUWA uses this private key to decrypt the encrypted PAN CSV data prior to processing.

Batch Upload Web Application PGP Public Encryption Keys

These are the public keys for test and live batch uploads.

Testing Key

```
-----BEGIN PGP PUBLIC KEY BLOCK-----
        Version: GnuPG v2.0.22 (GNU/Linux)
        mQENBF1AZqsBCADPhdh+mHeFZXwJnjbodJB5eKKkJh2islbwlvvU8VSqFTXdsGH0
        +6k3lGneNFwYrLK+GPBithbDtyWvvge9pZHm3Uz9szW475OtAqNo2x77d3jGE8i7
        e0siraK0e605DyimLBC0RFQxh9zF9e2D+5puIZwTVghrlraLlwge3JfkT684Iuos
        2VY22lKPH/gAUa4suqIwiOXmjyEFoFDIq0q+BrNr4OBqpTmbb8BL7hShxD9xWhMT
        FeamesWWIHRupRaEMDqDQftg48KsOJy0sPusHI6+cvyU9o6mJfp4Qt8fmYA5Mahv
        qVcNoP3TpmAtOJ8sgy9XL952WEi1GnekEy6XABEBAAG0JmN5YnNjYXNiYXRjaCA8
        Y3licy1kZXZvcHMtbXdAdmlzYS5jb20+iQE5BBMBAgAjBQJdQGarAhsBBwsJCAcD
        AgEGFQgCCQoLBBYCAwECHgECF4AACgkQJgDf08OgHZft2gf/bFW/9kM8enBss/bi
        q5jfhp27HRgdLW0q6IKmKtcdfXR0UTv3TBMVQGXJBbDzzKEIC9gUb3tnCmPYTHa3
        W8ikAHiSVdKY6I4Oo1395WQfzhPevvOl7q58MquX6eAfHs/urMzjGvtyc4MsoWR9
        njcnz4liCyd74cnas04Pu7eVE8ccI7Vc0qOQpBl66wPAguSvCOqgs5OW5HUqK0e3
        4nidXP79qy4bMsZgyKZz7byD/9fM7pfybLfDsvuMmRZIXmkmNVqB+UjkbHWRzZX5
        up2j9b7yE2vm4r4liMOicqXDyXs16K2ptsA23A2kwTJ4JMCh05K1Yu+pZmJ0D0ov
        gVkmh7kBDQRdQGdYAQgA3pkLZqqbSQhNPJ+gXOYjYXMUdumYvfSPI7XNN4wXxdeY
        vrxtMWrXjerEVyaioqJ0IRgfcxzMaAPL4AukbPaXRfCfC4HXh3AkD7FwB1pSnNk3
        qs+8e3sDZ50LkT9R1gFHYJ1d5nDwYqKHSVusgszmnGRCDVMNxEh8KePQGt9JaUqP
        2dA7kYAPondb09RBFwAoA/fdicfLo/1u24cXBJOg1OH3bGfjeuSC8gQoFG9nYwvP
        K9ZIjXXfWRDjdauI7Q9AUDfUWyDsZlA/KhemY72V6fB2OeBFzdOvL6C5Y/EaulxC
        RC5YHGKUxnbGGjT9T0HnMk/R7izHvPmNR578hv4AYwARAQABiQEfBBgBAgAJBQJd
        QGdYAhsMAAoJECYA39PDoB2XbasH/0E4ohVmJr9HgUE1n/p91PxbChK9uog5Wwke
        cCQKuF/qVPkoKPJkHlZBWLEAvQOBGXTXAXrNn1sA/xdKefS5jUSdMhoXt4EMykVH
        NyQgj2v/Qhh0y+nOiq9xFRmOurtrGyE2hFJuyMldPEJ9vyvqdBz2Iz0SDwIrbA88
        Ed5lCJ9Kdh1MyFk7xJE1aVl0s7JCLf9JUcID46rF2p88CEVmuwycaH3GVGgpy2O+
        STxFIce3PU12HKy28sQotEHWrKWRhRV6cldFswE4axeu3T4lqBXmGjDlKIXp8hlq
        2OR7fc12Kb77dzPE/EsriYTO5EbgCkFVUmQsXQI93H53glQYimk=
        =sYMU
        -----END PGP PUBLIC KEY BLOCK-----
```

Production Key

```
-----BEGIN PGP PUBLIC KEY BLOCK-----
        Version: GnuPG v2.0.22 (GNU/Linux)
        mQENBF1AankBCACuWaeJ9qj2eB/c6Ag+iRR5z6QmHDGXJ0flHm23WyH3gti3GETo
        ZjqRS3HMNt77HdW97WH9xWIFVytC3+VqAEPyZkhLUh5RMGWF5hLCl95rolMKBz6Q
        D/LlYcfp+nc8Nwb+SZSsi8S72Mk8djhJRYR79N6RJg1FzqmXqIqljRhuHWaZaDIT
        vAixS3cYkH2f7PlhgfUBnpyj6qeU8VMx5aaycjH6mzyOshc0lg4ojPH62GYvDK47
        caOoBCpmXxGCqx3tMLrf/wKk1vy8bccPFs2Nz7qgbNJkK8tbbN053/jV7ADnZKJc
        pJoz1H3DXbWn0x2CJAx64cNbhJ3zhZRyrveZABEBAAG0J2N5YnNwcm9kYmF0Y2gg
        PGN5YnMtZGV2b3BzLW13QHZpc2EuY29tPokBOQQTAQIAIwUCXUBqeQIbAQcLCQgH
        AwIBBhUIAgkKCwQWAgMBAh4BAheAAAoJEP36WF1kQenwmjQH/2daVD/GEIVCQxC6
        5XKER5YTj/G9UCSQXlxlrvDp/o8oVlJ5IgWeV6Sq9bWcYZLl4YPpgOXJEFDae8ne
        FuDaouiOO6UuupR3pGpu1LN4Zjzm5amkcqYPRURHNQmF5yRwTRCOd7/cJ+vy/uDP
        k4/s1l+qTqD16/dc9DIgW0B4OWvDJUEwRtlRasTLIa0doTePFD9mCrz8JQfvcD9X
        GDMjfaGb/WzG6s+4coghQR0gJWA/39k1V37m56kTghKxErMsL2929oMj5HWGJSI/
        xeinxkBOIyRN+367DH38V0Ux5IP0/jQacF0/i4UMv7uEcx0FecSMIfWuIi6GAnIK
        Tx8etYq5AQ0EXUBqsAEIAM3DfkjeX8b3OBVxBduHhRShUmAaN7JbkfquKd8fz5Ir
        iIAx+5MfZlie6AOorDnDyfDZAHqsDekLojbDOquYuVf2bFmpyDUuGoia2OTeA/pi
        ZIkvqa7NHkldpOzUcA56aDbwIs+Yv+b2iZNwDhYXUPyigOIvDADQG2LrliXICywS
        ecDw/nACnuyw3QvnFF8x06QSVINf4OiVq/CE5G/cPUgj7mpVjiGyBfkMnQkpeQz5
        /QGfox2mhn90kz6HcfHLmz+AAeH5ZNgnSEhKhgt/4BXjWFcLlhzwCvI+QVHGMmrz
        BcrUcVxSvBKHo0NgXECgxozuKJa2FUiydCVgJYXHCQcAEQEAAYkBHwQYAQIACQUC
        XUBqsAIbDAAKCRD9+lhdZEHp8PFYCACLdX+oj82lUw25kJvDn3Vils/iNlVTd9kM
        NLwQ3NbG/wYUDh8Nv5nN5sw8R0+ydIUJKm2NSxW++i7TWUtTn9a9hJPMf1svt1hX
        BrHODcEKDJQsBR+ITj7C+K1n6RVbCQjDdHannXUtD5UgnESTsuCQeDpMoVG4QR2g
        Wir1I2C2dcVtIKfl579ENgIRSmiL4KYldSsm7sXlleIhPEWOhGOP69i4iBkKOtsF
        oMQ/EWN1FiJeu9J0QyQMH1YlmXlPC63uQvep4AsdgSTYcZv4MnCnOuCKMFp7MZ4N
        3hcLs4kNqJDLZfTwupV8xfYd5eH9oX1tpBXnOc3LapNLL7zTsL6L
        =M9l9
        -----END PGP PUBLIC KEY BLOCK-----
```

Merchant PGP Key Pair

This key pair is used for request signature verification and to encrypt and decrypt reporting data. Generate this key pair and submit the public key for use by Account Updater through the online banking portal, by selecting the Merchant Services section and then click **Key Management**. For information about creating the merchant file signature PGP key pairs, see *Creating and Using Security Keys* (PDF | HTML).

- **Public key**: submit this key to Account Updater for two purposes:
    - Verify the signature of encrypted PAN CSV data prior to processing.
    - Encrypt the CSV report file containing PAN updates.
- **Private key**: use this key to sign the encrypted PAN CSV data and to decrypt the CSV report file that contains PAN updates.

Formatting a Request File

Account Updater request files must be in CSV format with a maximum file size of 10 MB. The format for a request file consists of:

- A header record.
- A detail record with one or more data records, each on a separate line.
- A footer record, which indicates the end of the file.

Request Header Record

The header record consists of comma-separated values and uses the fields listed in the following table:

Header Record Fields

Field Name	Description	Required or Optional	Data Type & Length
Record Identifier	Constant value indicating the record type. Format: H	Required	Alpha (1)
File Classification	Indicates whether this is a request or response file. Format: cybs.au.request.pan	Required	Alpha (30)
merchantID	Your merchant ID. Format: sampleID2	Required	Alphanumeric (30)
batchID	File (batch) identifier that you assign. The batch ID must be unique. If you send a file that contains a previously submitted batch ID, the file is rejected. Format: 12345	Required	Numeric (30)
recordCount	The number of detail records in the file. Format: 12345	Required	Numeric
statusEmail	Email address to which status emails for the request are sent. Format: aaa@aaa.aaa	Required	Alphanumeric (100)

Header Record Fields (continued)

Field Name	Description	Required or Optional	Data Type & Length
creationDate	Optional field that you can pass for reference. If present, it appears in the online banking portal at Merchant Services, then Account Updater View Status window. Format: yyyy-MM-DD	Optional	String (10)
Batch Info	Optional field that you can pass for reference. Format: sample12	Optional	Alphanumeric (50)

Request Detail Record

Each file must contain at least one detail record.

Detail Record Fields

Field Name	Description	Required or Optional	Data Type & Length
Record Identifier	Constant value indicating the record type. Format: D	Required	Alpha (1)
Card Number	Card number to process.	Required	Numeric (19)
Card Expiration Month	Expiration month of the card. Format: MM	Required	Alphanumeric (2)
Card Expiration Year	Expiration year of the card. Format: YY	Required	Numeric (2)
Merchant Reference ID	You can use this field to track your Account Updater request records. If this field is populated, the same value is returned in the Account Updater response file. Format: sampleID2	Optional	Alphanumeric (50)

Detail Record Fields (continued)

Field Name	Description	Required or Optional	Data Type & Length
BA Sub Merchant ID	This field is required for billing aggregator merchants only. Format: sampleID2	Optional	Alphanumeric (10)

Request Footer Record

Each file should contain only one footer record.

Footer Record Field

Field Name	Description	Required or Optional	Data Type & Length
Record Identifier	Constant value indicating the record type. Format: F	Required	Alpha (1)

Request File Examples

Example: Non-Billing Aggregator Merchants

```
H,cybs.au.request.pan,merchant1,001,2,notify@yourcompany.com,2019-03-23,My January Batch
D,1111222233334444,11,09,0001
D,2222333344445555,11,09,0002
      F
```

Uploading a Request File

> **Important:** For each PAN you upload, you can receive multiple responses. For example, if you upload one Visa card for an update, you can receive both a Mastercard and Visa response, or two Visa responses.

To upload the request file, use HTTPS. Your client application must support HTTP/1.0 or HTTP/1.1 and TLS 1.2 or later.

To access the Account Updater URL, you must provide the same Simple Order API client certificate that you use to request regular individual ICS Simple Order API transactions. The client certificate is stored in a PKCS12 file named *<merchantID>.p12* and is protected by a single password.

Before you submit files to the production server, test your request files. Follow the instructions in Testing (on page 42).

Use the following URLs for submitting test and live Account Updater request files:

- **Test:** https://accountupdatertest.merchant-services.bankofamerica.com/upload/encrypted-pans
- **Production:** https://accountupdater.merchant-services.bankofamerica.com/upload/encrypted-pans

The request is a POST form data request with the encrypted file keyed as a data file.

The request requires a basic authorization header containing your merchant ID and the merchant signature PGP public key fingerprint colon delimited and Base64 encoded. For example:

```
Authorization: Basic
 base64Encode(<merchant-id>:<hex-formatted-merchant-public-key-fingerprint>)
```

A successful request results in an HTTP 200 response code. The following error codes are possible:

HTTP Error Codes

Code	Description	Failure Scenarios
400	Bad Request	Malformed request. Payload too large.
401	Unauthorized	Invalid request credentials. Public key matching public key fingerprint not found.
403	Forbidden	Invalid merchant header in uploaded file.
500	Internal Server Error	Message fails to be put on queue for processing.
503	Server Unavailable	Internal database unavailable to retrieve public key.

See Sample Java Code for Uploading PANs (on page 59) for more information on creating a client certificate to upload request files.

Email Notification

After you upload the request file, Account Updater validates the syntax and sends you a confirmation email indicating whether the file passed this stage of validation. You must specify an email address in the **statusEmail** header field in order to receive this confirmation email. If this field is left blank, you do not receive an email confirmation, and you must go to Tools found by logging into Business Advantage 360, selecting Merchant Services, and then Account Updater to view the status. Account Updater sends the email notification within 30 minutes of receiving the request file. However, actual timing depends on the system load when the file is submitted.

The table below lists possible subject lines of the email notifications.

Email Notifications

Subject Line	Reason	Status Viewable in the Online Banking Merchant Services
Received	The Account Updater request file was received. Account Updater processes the requests in the file. No action is required.	Yes
Rejected	The file was rejected. Read the contents of the email and follow the suggested remedy.	No
Validated	The file passed validation.	Yes
Declined	The file did not pass validation checks. All records are declined. Read the contents of the email and follow the suggested remedy.	Yes
Processing	The request file is being processed by Account Updater.	Yes
Completed	The response file has been generated and is ready for download.	Yes

Related information

Viewing the Batch File Status *(on page 38)*

Viewing the Status of a Batch File

1. Log in to your Bank of America Business Advantage 360 account.
2. Select **Merchant Services**.
3. On the left navigation pane, click the **Tools** icon.
4. Click **Account Updater**. The Account Updater page appears.
5. Use the filters on the Search toolbar to find the batch you want to view. The Search Results list shows matching results.

Downloading a Response File

You can download response files with a status of *Complete* from Merchant Services with your account or with a client application. To download it programmatically, see Secure File Share API at the Developer Center.

1. Log in to your Bank of America Business Advantage 360 account.
2. Select **Merchant Services**.
3. On the left navigation pane, click the **Reporting** icon.
4. Under Downloadable Reports, click **Report List**. The Report List page appears.
5. Click the Third-Party Reports tab. The Third-Party Reports page appears.
6. In the Download column, click the file format link.
   Only reports that have successfully finished generating and that contain data include links.
7. Follow your browser's instructions to open and save the file.

Response File Records

The response file is encrypted with the public part of the PGP Key that you generated and uploaded to Account Updater. To read a response file, you must decrypt it using the private part of the PGP key pair. You can do so with the same third-party software you used to create the keys.

The format for a request file consists of:

- A header record.
- A detail record with one or more data records, each on a separate line.
- A footer record, which indicates the end of the file.

Response Header Record

The header record consists of comma-separated values and uses the fields listed in the following table:

Header Record Fields

Field Name	Description	Data Type & Length
Record Identifier	Constant value indicating the record type. Format: H	Alpha (1)
File Classification	Indicates whether this is a request or response file, and the type of service. Format: cybs.au.response.pan	Alphanumeric (30)
MerchantID	Your merchant ID.	Alphanumeric (30)
BatchID	File (batch) identifier sent in the request file.	Numeric (30)

Response Detail Record

Each file contains at least one detail record.

Response Detail Record Fields

Field Name	Description	Data Type & Length
Record Identifier	Constant value indicating the record type. Format: D	Alpha (1)
Request ID	Unique identifier for the record.	Numeric (30)
Old Card Number	Old card number.	Numeric (19)
Old Card Expiration Month	Old expiration month. Format: MM	Numeric (2)
Old Card Expiration Year	Old expiration year. Format: YY	Numeric (2)
New Card Number	New card number.	Numeric (19)
New Card Expiration Month	New expiration month. Format: MM	Numeric (2)
New Card Expiration Year	New expiration year. Format: YY	Numeric (2)
Merchant Reference ID	This field is optional and is returned in the response if present in the request file.	Alphanumeric (50)
BA Sub Merchant ID	This field is returned in the response if sent in the request file.	Alphanumeric (10)
Response Code	Response code for the record.	Alpha (3)
Reason Code	Reason code for the record.	Numeric (3)

Related information

PAN Upload Response Codes and Reason Codes *(on page 55)*

Response Footer Record

Each file contains only one footer record.

Footer Record Fields

Field Name	Description	Data Type & Length
Record Identifier	Constant value indicating the record type. Format: F	Alpha (1)
Record Count	The number of detail records in the file.	Numeric (10)
Response Code	Response code for the file.	Alpha (3)
Reason Code	Reason code for the file.	Numeric (3)

Related information

PAN Upload Response Codes and Reason Codes *(on page 55)*

Response File Examples

Example: Non-Billing Aggregator Response File

```
H,cybs.au.response.pan,merchant1,001
D,1000000000000000001,1111222233334444,11,09,,,,0001,,NUP,800
D,1000000000000000002,2222333344445555,11,09,6666777788889999,11,11,0002,,NAN,800
F,2,COM,800
```

Testing

The Account Updater test environment provides a simulator in which the response from the card association can be triggered using card numbers listed in the following sections:

- American Express Test Card Numbers (on page 42)
- Mastercard Test Card Numbers (on page 43)
- Visa Test Card Numbers (on page 44)

This simulator ensures that you can handle the possible response combinations when connecting to multiple card associations.

The test environment typically completes the process in a matter of minutes rather than the 24-hour (or longer) duration of the live environment when updates are sent to the actual card associations.

American Express Test Card Numbers

American Express card updates through the Cardrefresher are available only if you are using Token Management. Use the numbers listed in the following tables to simulate various scenarios. Replace the BIN with `371449` and remove spaces when sending to Account Updater.

American Express Test Numbers

Card Number	Response Code
BIN `0 0002 0115`	NAN (No New Expiry Date)
BIN `1 0211 2216`	NAN (No New Expiry Date)
BIN `2 0121 2206`	NAN (No New Expiry Date)
BIN `1 0021 1119`	NAN (No New Expiry Date)
BIN `1 0101 0023`	NAN (No New Expiry Date)
BIN `0 0100 2112`	NAN (New Expiry Date)
BIN `1 2101 2009`	NAN (New Expiry Date)
BIN `0 2201 2009`	NAN (New Expiry Date)
BIN `2 1000 0113`	NAN (New Expiry Date)
BIN `0 2100 0229`	NAN (New Expiry Date)
BIN `2 2210 0224`	NED

American Express Test Numbers (continued)

Card Number	Response Code
BIN `0 0112 0203`	NED
BIN `0 2102 1100`	NED
BIN `2 0121 2107`	NED
BIN `0 1121 0119`	NED
BIN `0 1022 1109`	ACL
BIN `1 0112 1226`	ACL
BIN `2 0201 0005`	ACL
BIN `1 2121 0207`	ACL
BIN `0 1012 2109`	ACL
BIN `1 2120 0224`	DEC
BIN `2 1010 0020`	861 (Attempt to enroll. Customer already enrolled.)
BIN `1 1202 1118`	862 (Registry rejected due to card member opt out.)
BIN `1 0122 0218`	ERR
BIN `0 0010 2004`	ERR
BIN `1 1111 2108`	861
BIN `1 4254 6639`	NAN (No New Account Number, No New Expiry Date)
BIN `6 7595 0950`	NAN (No New Account Number, New Expiry Date)

Related information

PAN Upload Response Codes and Reason Codes *(on page 55)*

Mastercard Test Card Numbers

The bold fields represent the token updates for Token Management, Recurring Billing, and Payment Tokenization merchants using the REST API batch update and harvest update. Replace the BIN with `511111` and remove spaces when sending to Account Updater.

Mastercard Card Test Numbers

Card Number	Response Codes
BIN `10 4714 3086`	**Visa Response: NAN** | Mastercard Response: NAN
BIN `10 2999 7178`	Visa Response: ACL |**Mastercard Response: NAN**

Mastercard Card Test Numbers (continued)

Card Number	Response Codes
BIN `10 1548 6814`	Visa Response: CUR |**Mastercard Response: NAN**
BIN `10 5459 2548`	Visa Response: NUP | **Mastercard Response: NAN**
BIN `10 4871 8571`	Visa Response: CCH | **Mastercard Response: NAN**
BIN `10 5798 7356`	**Visa Response: NAN** | Mastercard Response: NED
BIN `10 7450 2964`	Visa Response: ACL | **Mastercard Response: NED**
BIN `10 6971 3154`	Visa Response: CUR |**Mastercard Response: NED**
BIN `10 2030 4416`	Visa Response: NUP |**Mastercard Response: NED**
BIN `10 4733 5823`	Visa Response: CCH | **Mastercard Response: NED**
BIN `10 3135 3600`	**Visa Response: NAN** | Mastercard Response: ACL
BIN `10 4816 3604`	**Visa Response: ACL** | Mastercard Response: ACL
BIN `10 1867 3020`	Visa Response: CUR |**Mastercard Response: ACL**
BIN `10 3056 0627`	Visa Response: NUP |**Mastercard Response: ACL**
BIN `10 0270 8865`	Visa Response: CCH |**Mastercard Response: ACL**
BIN `10 6646 9396`	**Visa Response: NAN** | Mastercard Response: CUR
BIN `10 5787 1816`	**Visa Response: ACL** | Mastercard Response: CUR
BIN `10 7350 8855`	**Visa Response: CCH** | Mastercard Response: CUR

Related information

PAN Upload Response Codes and Reason Codes *(on page 55)*

Visa Test Card Numbers

The bold response codes represent the token updates for Token Management, Recurring Billing, and Payment Tokenization merchants using the REST API batch update and harvest update. Replace the BIN with `400000` and remove spaces when sending to Account Updater.

Visa Card Test Numbers

Card Number	Response Codes
BIN `71 0951 9220`	Visa Response: NAN |**Mastercard Response: NAN**
BIN `15 3919 2096`	**Visa Response: NAN** | Mastercard Response: ACL
BIN `18 6481 0239`	**Visa Response: NAN** | Mastercard Response: CUR

Visa Card Test Numbers (continued)

Card Number	Response Codes
BIN `91 9582 8465`	Visa Response: NED |**Mastercard Response: NAN**
BIN `27 5765 7455`	**Visa Response: NED** | Mastercard Response: ACL
BIN `71 1311 2087`	**Visa Response: NED** | Mastercard Response: CUR
BIN `21 1752 4874`	Visa Response: ACL |**Mastercard Response: NAN**
BIN `71 1629 4650`	Visa Response: ACL |**Mastercard Response: ACL**
BIN `20 5548 7183`	**Visa Response: ACL** | Mastercard Response: CUR
BIN `52 8063 4792`	Visa Response: CUR |**Mastercard Response: NAN**
BIN `24 0631 2635`	Visa Response: CUR |**Mastercard Response: ACL**
BIN `89 2339 9344`	Visa Response: CUR |**Mastercard Response: CUR**
BIN `55 7908 8940`	Visa Response: NUP |**Mastercard Response: NAN**
BIN `57 9875 5634`	Visa Response: NUP |**Mastercard Response: ACL**
BIN `80 9110 0706`	Visa Response: CCH |**Mastercard Response: NAN**
BIN `26 9567 5155`	Visa Response: CCH |**Mastercard Response: ACL**
BIN `35 8627 6236`	**Visa Response: CCH** | Mastercard Response: CUR

Related information

PAN Upload Response Codes and Reason Codes *(on page 55)*

API Fields

The following fields can be used with Account Updater.

Request Fields

Request Fields

Field Name	Description	Used By & Required (R) / Optional (O)	Validation
notificationEmail	Email address to which batch status updates are sent.	POSTs to /batches (R)	Valid email address
merchantReference	Your reference to identify the batch.	POSTs to /batches (O)	0 to 255 characters
type	Indicates whether batch is a one-off update for Visa and/or Mastercard, or an enrollment in American Express Cardrefresher. Possible values: - `oneOff` (default): Visa or Mastercard - `amexRegistration`: American Express	POSTs to /batches (O)	
included	Elements to be included. Must include one of the following: - `tokens` - `instrument_identifier`	POSTs to /batches (R)	

Request Fields (continued)

Field Name	Description	Used By & Required (R) / Optional (O)	Validation
tokens	Comma-separated list of subscription IDs, Token Management *customer* or *payment instrument* tokens.	POSTs to /batches (O)	If the array is present, then it should not be empty (min length = 1) or contain null values. Maximum number of tokens is 10 million.
instrumentIdentifiers	Token Management *instrument identifier* token assigned to the tokenized PAN and its associated expiration dates.	POSTs to /batches (O)	
id	ID for the *instrument identifier* token.	POSTs to /batches (R)	String (32)
expirationMonth	Two-digit month in which the card expires.	POSTs to /batches (R)	String (2)
expirationYear	Four-digit year in which the card expires.	POSTs to /batches (R)	String (4)

Response Fields

Response Fields

Field Name	Description	Returned By	Data Type & Length
batchId	When the request is successful, a batch ID is returned to the user.	/batches /…/status /…/report	Alphanumeric (26)
batchItemCount	When the request is successful, this value is the number of items that were included in the request. When the request is unsuccessful, the value of this field is `0`.	/batches /…/status /…/report	Numeric (9)
_links	JSON object containing link elements relating to the request. Successful requests return the URI of the batch status.	/batches /…/status /…/report	
self	The resource address that was requested. Element within **_links**.	/batches /…/status /…/report	URL
first	First page in the result set.	/batches	URL
next	Next page in the result set.	/batches	URL
last	Last page in the result set.	/batches	URL
status	URI of the batch status resource. ! **Important:** Do not hard-code the link to the batch status resource. Use the returned value to avoid errors if the URI structure changes.	/batches	URL

Response Fields (continued)

Field Name	Description	Returned By	Data Type & Length
reports	URI of the batch associated with the **batchId**.	/batches /status	URL
correlationId	Returned when an error occurs. Provide this ID to Customer Support to help identify your transaction.	/batches /…/status /…/report	String (36)
code	HTTP Response code. Returned when an error occurs.	/batches /…/status /…/report	String (3)
detail	Returned when an error occurs. Detailed description of the error.	/batches /…/status /…/report	String (1024)
fields	Returned when an error occurs. The array contains elements that describe the erroneous fields.	/batches	
path	Returned when an error occurs. Element within the fields. Path of field name.	/batches	String (36)
message	Returned when an error occurs. This is a plain text error message and can be an element within the fields. This field can also appear with the fields JSON object.	/batches	String (256)
localizationKey	Returned when an error occurs. A unique key that represents the error message and can be an element within fields. This field can also appear with the JSON object.	/batches	String (128)

Response Fields (continued)

Field Name	Description	Returned By	Data Type & Length
version	Version of the report. For example, v1.4-1 is the major version of the API used to create the batch and 4 is the minor version of the report. You always receive the latest minor version of the report for the API you used to create the batch.	/report	

batchSource	Method used to create the batch. For example, `TOKEN_API`.	/batches
		/status
		/report
			`TOKEN_API`
			`SCHEDULER`
			`AMEX_REGISTRY`
			`AMEX_MAINTENANCE`

batchCaEndpoints	Card associations to which the card numbers were sent.	/batches /status /report	Array containing one or more of the following: - `VISA` - `MASTERCARD` - batchCaEndpoints `AMEX`
batchCreatedDate	Date on which the batch was created.	/batches /status /report	ISO_8601 UTC date
reportCreatedDate	Date on which the report was created.	/batches /status /report	ISO_8601 UTC date

Response Fields (continued)

Field Name	Description	Returned By	Data Type & Length
merchantReference	Your reference, if present in the request.	/batches /status /report	0 to 255 characters
totals	JSON object containing the high-level summary of the batch.	/batches /status /report	
acceptedRecords	Number of tokens that were identified and retrieved for the merchant ID.	/batches /status /report	String (9)
rejectedRecords	Number of tokens that were not identified and retrieved.	/batches /status /report	String (9)
updatedRecords	Number of updates that were applied to a token.	/batches /status /report	String (9)
caResponses	Number of updates that were received from the card associations. This value represents updates that may have or have not been applied to a token.	/batches /status /report	String (9)
caResponsesOmitted	Number of updates that were not applied to a token. For example, a response is returned by more than one card association.	/batches /status /report	String (9)
billing	JSON object containing the billing summary information.	/status /report	

Response Fields (continued)

Field Name	Description	Returned By	Data Type & Length
nan/ned/acl/cch	Number of each billed response type.	/status /report	String (3)
records	JSON object containing additional objects that relate to the original tokens and the updates or errors that occurred.	/report	
id	ID for the record.	/report	
sourceRecord	JSON object containing details from the source token.	/report	
token	Subscription ID included in the request.	/report	
cardNumber	Masked card number before an update. First six digits and the last four digits are not masked.	/report	
cardExpiryMonth	Two-digit month in which the card expires.	/report	
cardExpiryYear	Four-digit year in which the card expires.	/report	
cardType	Type of card. Possible values: - `001`: Visa - `002`: Mastercard - `003`: American Express	/report	String (3)
customerId	Value of the *customer* token assigned to the tokenized shipping information and merchant defined data. This field is for Token Management merchants only.	/report	

Response Fields (continued)

Field Name	Description	Returned By	Data Type & Length
paymentInstrumentId	The value of the *payment instrument* token assigned to the tokenized billing information and card expiration dates. This field is for Token Management merchants only.	/report	
instrumentIdentifierId	Value of the *instrument identifier* token assigned to the tokenized PAN. This field is for Token Management merchants only.	/report	String (32)
responseRecord	JSON object containing the details that were made to the token.	/report	
response	Type of response.	/report	
reason	Reason code for the response.	/report	
token	If last-four-digit format-preserving tokens are used, a new token (subscription ID) can be returned that replaces the source record token.	/report	
cardNumber	Masked card number. First six and last four digits are not masked.	/report	
cardExpiryMonth	Two-digit month in which the card expires.	/report	
cardExpiryYear	Four-digit year in which the card expires.	/report	
cardType	Type of card. Possible values: - `001`: Visa - `002`: Mastercard - `003`: American Express	/report	String (3)

Response Fields (continued)

Field Name	Description	Returned By	Data Type & Length
instrumentIdentifierId	Value of the *instrument identifier* token assigned to the updated tokenized PAN. This field is for Token Management merchants only.	/report	String (32)
instrumentIdentifierId Created	Indicates whether this is the first time the PAN has been tokenized for you. Possible values: - `true` - `false`	/report	String (5)
additionalUpdates	Details associated with a closed *instrument identifier* token.	/report	
customerId	Value of the *customer* token not present in the batch and associated with a closed *instrument identifier* token.	/report	String
paymentInstrumentId	Value of the *payment instrument* token not present in the batch and associated with a closed *instrument identifier* token.	/report	String
creator	Merchant that created the *payment instrument* token	/report	String
state	State of the token.	/report	“CLOSED”
message	Information about the tokens.	/report	String

Related information

PAN Upload Response Codes and Reason Codes *(on page 55)*

PAN Upload Response Codes and Reason Codes

Record Level

The response code and the reason code for the record appear in the details record of the request file.

Example: Details Record

```
D,1000000000000000002,2222333344445555,11,09,6666777788889999,11,11,0002,,NAN,800
```

Response Codes and Reason Codes

Response Code	Response Code Description	Reason Code	Reason Code Description	Billable or Non-Billable Code
ACL	Match: account closed. The status of the customer subscription changes to cancelled and all recurring billing payments stop.	800	Success.	Billable.
CCH	Contact card holder.	800	Success.	Billable.
CUR	Card data current.	800	Success.	Non-billable.
DEC	—	801	Invalid card number.	Non-billable.
DEC	—	802	Invalid check digit.	Non-billable.
DEC	—	803	Invalid expiration date.	Non-billable.
DEC	—	804	Unsupported card type.	Non-billable.
DEC	—	805	Invalid card type length.	Non-billable.
DEC	—	806	Unknown card type.	Non-billable.

Response Codes and Reason Codes (continued)

Response Code	Response Code Description	Reason Code	Reason Code Description	Billable or Non-Billable Code
DEC	—	810	Invalid BA sub merchant ID.	Non-billable.
DEC	—	850	Invalid token format.	Non-billable.
DEC	—	851	Invalid token length.	Non-billable.
DEC	—	852	Unknown token. This token does not exist, is not associated with your account, or might be superseded.	Non-billable.
DEC	—	853	Invalid token status. This token has a status of CLOSED from a previous Account Updater batch.	Non-billable.
DEC	—	861	Cardholder is already enrolled or cannot cancel cardholder that is not enrolled.	Non-billable.
DEC	—	862	Rejected because cardholder opted out.	Non-billable.
ERR	—	801	Invalid card number.	Non-billable.
ERR	—	802	Invalid check digit.	Non-billable.
ERR	—	803	Invalid expiration date.	Non-billable.
ERR	—	804	Unsupported card type or cancelled card.	Non-billable.
ERR	—	807	Merchant not enrolled properly in Account Updater.	Non-billable.

Response Codes and Reason Codes (continued)

Response Code	Response Code Description	Reason Code	Reason Code Description	Billable or Non-Billable Code
ERR	—	808	Incorrect record indicator.	Non-billable.
ERR	—	809	Unknown error code received during processing.	Non-billable.
ERR	—	811	New account number failed MOD-10 check.	Non-billable.
NAN	New account number. It might also include a new expiration date.	800	Success.	Billable.
NED	New expiration date.	800	Success.	Billable.
NUP	No match, no update.	800	Success.	Non-billable.
UNA	Inconsistent update received, not applicable.	800	Inconsistent update received, not applicable.	Non-billable.

Request File Level

The response code and the reason code for the request file appear in the footer record of the request file.

Example: Footer Record

```
F,2,COM,800
```

Response Codes and Reason Codes

Response Code	Response Code Description	Reason Code	Reason Code Description
COM	The merchant request file has been validated, processed, and the response received.	800	Success.

Response Codes and Reason Codes (continued)

Response Code	Response Code Description	Reason Code	Reason Code Description
DEC	The merchant request file was not processed because each record failed record-level validation.	801	All records within the request file failed record-level validation.

Sample Java Code for Uploading PANs

This section explains how to use the sample Java code to upload your files to Account Updater.

Requirements

- J2SE 1.5 or later.
- Unlimited Strength Jurisdiction Policy files from Oracle (*US_export_policy.jar* and *local_policy.jar*):

  http://www.oracle.com/technetwork/java/javase/documentation/index.html

- Bouncy Castle, which includes *bcmail*.jar*, *bcpg*.jar*, *bcprov*.jar*, and *bctest*.jar*:

  www.bouncycastle.org

Using the Sample Code

The sample code was developed and tested on a Solaris platform.

1. Replace your Java installation's existing security policy files with the new ones you downloaded from Oracle's site:

   a. Find your existing *US_export_policy.jar* and *local_policy.jar* files in the *$JAVA_HOME/jre/lib/security* directory.

   b. Rename or move your existing files to another directory.

   c. Copy the new *US_export_policy.jar* and *local_policy.jar* files that you downloaded from Oracle to the *$JAVA_HOME/jre/lib/security* directory.

2. Copy the Bouncy Castle **.jar* files to the *$JAVA_HOME/jre/lib/ext* directory.

3. Edit the *$JAVA_HOME/jre/lib/security/java.security* file and insert the security provider immediately after the Oracle provider. Be sure to increment the numbers of the other providers in the list.

4. Insert this line: `Security.provider.2=org.bouncycastle.jce.provider.BouncyCastleProvider`
   Your list of security providers should now look like this:

```
security.provider.1=sun.security.provider.Sun
security.provider.2=org.bouncycastle.jce.provider.BouncyCastleProvider
security.provider.3=com.sun.net.ssl.internal.ssl.Provider
security.provider.4=com.sun.rsajca.Provider
security.provider.5=com.sun.crypto.provider.SunJCE
security.provider.6=sun.security.jgss.SunProvider
```

5. Import your Simple Order API .p12 security key into Internet Explorer:

   a. Open Internet Explorer, and choose **Tools > Internet Options**.

   b. Click the **Content** tab.

   c. Click **Certificates**.

   d. Click **Import** to open the Certificate Import Wizard, and click **Next** to start the Wizard.

   e. Browse to the location of your .p12 security key, and click **Next**.

   f. For the password for the private key, enter your merchant ID. For example, if your key is giraffe.p12, enter `giraffe` as the password.

   g. On this page, check the box for **Mark this key as exportable**, and click **Next**.

   h. Click **Next** on the Certificate Store page.

   i. Click **Finish**. A confirmation message appears indicating that the import was successful.

6. Create a key store file to contain your Simple Order API .p12 security key:

   a. Browse to one of the following URLs:

   Test: https://accountupdatertest.merchant-services.bankofamerica.com/upload/UploadAccountUpdaterFile

   Production: https://accountupdater.merchant-services.bankofamerica.com/upload/UploadAccountUpdaterFile

   b. Choose **File > Properties**.

   c. Click **Certificates**.

   d. Click the **Certification Path** tab.

   e. Click **Entrust.net Secure Server Certification Authority**.

   f. Click **View Certificate**.

g. Click the **Details** tab.

h. Click **Copy to File** and then **Next**.

i. Click **Browse** and navigate to a location to save the file.

j. Enter a name for the file, such as *MyCert*. Click **Save** and click **Next**.

k. Click **Finish**. Your file (*MyCert.cer*) has been created in the location you specified.

l. Go to the *$JAVA_HOME/bin/keytool* file and use the J2SE keytool program to create a keystore file that contains this newly created certificate. You must provide a pass phrase for the keystore. You MUST use the same password that you used in Step 5. For example, if your p12 key is giraffe.p12, the pass phrase must be *giraffe*.

m. To create the keystore, enter this command: **$JAVA_HOME/bin/keytool -import -file <path to certificate>/<name of certificate file> -keystore <name of keystore file>.jks -storepass <pass phrase of keystore>**
Example Request: Creating the Keystore

```
$JAVA_HOME/bin/keytool -import -file /home/bluu/MyCert.cer-keystore
 MyKeystore.jks -storepass myMerchantID
```

The output looks like this example:

Example Response: Creating the Keystore

```
Owner: CN=accountupdatertest.merchant-services.bankofamerica.com,
 OU=Operations, O=Your Company, L=Your City, ST=California, C=US Issuer:
 CN=Entrust.net Secure Server Certification Authority, OU=(c) 1999
 Entrust.net Limited, OU=www.entrust.net/CPS incorp. by ref. (limits
 liab.), O=Entrust.net, C=US Serial number: 374e1b7b Valid from: Thu Nov
 18 17:15:34 PST 2018 until: Tue Jan 31 17:51:24 PST 2020 Certificate
 fingerprints: MD5: BE:BF:B0:91:69:C4:7B:10:45:EC:D6:0F:16:AA:3D:77
 SHA1: 07:F8:41:DC:B2:FC:F5:DA:FC:EE:09:7A:33:B8:29:15:31:18 Trust this
 certificate? [no]: yes Certificate was added to keystore
```

7. Modify the *SSLFileTransfer.props* file with your settings. The file is part of the download package and looks similar to this example:
Example: Modifying the SSLFileTransfer.props File

```
># Upload host host=accountupdatertest.merchant-services.bankofamerica.com
 # Upload port port=<upload port> # Username to log into the Business
 Center bcUserName=<Business Center login name> # Password to log into the
 Business Center bcPassword=<Business Center login password> # File to upload
 uploadFile=<path to your file>/<file name> # Path where to upload the file
 (provided by Bank of America) path=/upload/UploadAccountUpdaterFile # Your
```

```
Bank of America security key key=<key location path>/<key file name> #
New key store you just created that contains the certificate keyStore=<key
store location>/<new key store name> # pass phrase is the string you
used in -storepass option when you # created the key store file earlier
passPhrase=<pass phrase>
```

8. Set the JAVA_HOME environment variable to the location in which you installed J2SE.
   Example: Java Home Environment

```
JAVA_HOME=/home/j2se
```

9. Include *$JAVA_HOME/bin* in the PATH.

10. Compile and run the sample:

    a. Change to the directory containing the sample files.

    b. Enter the following:

    javac SSLFileTransfer.java

    java SSLFileTransfer <path to props file>**/SSLFileTransfer.props**

    If the upload is successful, the output will look similar to this example:

    Example: Upload Response

```
HTTP/1.1 200 OK
Date: Wed, 26 Jan 2005 17:26:31 GMT
Server: Apache Coyote/1.0
Content-Type: text/plain
Content-Length: 0
X-Cache: MISS from <your host>
Connection: close
UPLOAD FILE SUCCESSFUL
```