Token Management Service

REST API

Developer Guide

cybersource
A Visa Solution

Version: 25.04.01

Contents

Token Management Service Developer Guide

This developer guide is written for merchants who want to tokenize customers' sensitive personal information and eliminate payment data from their networks to ensure that it is not compromised. The purpose of this guide is to help you create and manage tokens.

Conventions

These special statements are used in this document:

Important: An *Important* statement contains information essential to successfully completing a task or learning a concept.

Warning: A *Warning* contains information or instructions, which, if not heeded, can result in a security risk, irreversible loss of data, or significant cost in time or revenue or both.

Related Documentation

Refer to the Technical Documentation Hub in the Cybersource Developer Center for additional technical documentation:

https://developer.cybersource.com/docs.html

Customer Support

For support information about any service, visit the Support Center:

http://support.visaacceptance.com

Recent Revisions to This Document

25.04.01

TMS Workflows

Updated the PAN tokenization workflow and steps. See PAN Tokenization Process Using TMS (on page 26).

25.01.01

Network Tokens

Added support for retrieving an AFT cryptogram as a network token payment credential. See Retrieve Network Token AFT Payment Credentials (on page 327).

Updated the endpoint for generating a network token for an existing instrument identifier. See Provision a Network Token for an Existing Instrument Identifier (on page 309).

TMS Workflows

Updated the MIT workflows for merchants and partners. See Network Token MIT for Merchants (on page 35) and Network Token MIT for Partners (on page 42).

Life-Cycle Management

Added examples for life-cycle management notifications after a network token update. See Network Token Life-Cycle Management (on page 72).

Added life-cycle management reason values. See Network Token Life-Cycle Management (on page 72).

Added a note about setting the organization ID to that of the vault owner when creating a new webhook subscription. See Create Webhook Subscription (on page 82).

24.11.01

Instrument Identifier Tokens

Added push provisioning to the list of supported features for instrument identifier tokens. See Instrument Identifier Tokens (on page 22).

Message-Level Encryption Keys

Added a statement about creating MLE keys for multiple merchants. See Message-Level Encryption Keys (on page 65).

Network Tokens

Added support for push provisioning for network tokens. See Push Provisioning for Network Tokens (on page 339).

Test Card numbers

Updated the list of test card numbers for network token provisioning. See Test Card Numbers (on page 56).

Token Requestor IDs

Added steps for entering the acquirer ID during token requestor ID enrollment. See Token Requestor IDs (on page 75).

Token Management Service Workflows

Added a workflow for push provisioning. See Push Provisioning Process (on page 28).

24.10.01

BIN Request

Added support for retrieving BIN details in a TMS request. See BIN Lookup Service and TMS (on page 348).

Network Tokens

Added support for deleting a standalone network token. See Delete a Standalone Network Token (on page 321).

HTTP Status Codes

Added the `502` HTTP status code. See HTTP Status Codes (on page 358).

24.09.01

Unmasked Payment Details

Added information on setting the header and configuring your settings for retrieving unmasked payment details. See these topics:

- Retrieve a Customer's Default Payment with an Unmasked Card Number (on page 101)
- Retrieve a Customer Payment Instrument with an Unmasked Card Number (on page 182)
- Retrieve a Payment Instrument with an Unmasked Card Number (on page 225)
- Retrieve an Instrument Identifier with an Unmasked Card Number (on page 267).

Network Tokens

Added support for provisioning a network token for a card and consumer ID. See Provision a Network Token for a Consumer (on page 312).

Added support for provisioning a network token for a token. See Provision a Network Token for a Token (on page 315).

Added support for retrieving a network token. See Retrieve a Standalone Network Token (on page 318).

Added `EXPIRED` as a network token status. See Network Token Life-Cycle Management (on page 72).

Message-Level Encryption Keys

Added information about extracting the public key. See Message-Level Encryption Keys (on page 65).

24.08.01

Test Card Numbers

Updated test card numbers for provisioning network tokens. See Test Card Numbers (on page 56).

VISA Platform Connect: Specifications and Conditions for Resellers/Partners

The following are specifications and conditions that apply to a Reseller/Partner enabling its merchants through Cybersource for Visa Platform Connect ("VPC") processing. Failure to meet any of the specifications and conditions below is subject to the liability provisions and indemnification obligations under Reseller/Partner's contract with Visa/Cybersource.

1. Before boarding merchants for payment processing on a VPC acquirer's connection, Reseller/Partner and the VPC acquirer must have a contract or other legal agreement that permits Reseller/Partner to enable its merchants to process payments with the acquirer through the dedicated VPC connection and/or traditional connection with such VPC acquirer.

2. Reseller/Partner is responsible for boarding and enabling its merchants in accordance with the terms of the contract or other legal agreement with the relevant VPC acquirer.

3. Reseller/Partner acknowledges and agrees that all considerations and fees associated with chargebacks, interchange downgrades, settlement issues, funding delays, and other processing related activities are strictly between Reseller and the relevant VPC acquirer.

4. Reseller/Partner acknowledges and agrees that the relevant VPC acquirer is responsible for payment processing issues, including but not limited to, transaction declines by network/issuer, decline rates, and interchange qualification, as may be agreed to or outlined in the contract or other legal agreement between Reseller/Partner and such VPC acquirer.

DISCLAIMER: NEITHER VISA NOR CYBERSOURCE WILL BE RESPONSIBLE OR LIABLE FOR ANY ERRORS OR OMISSIONS BY THE VISA PLATFORM CONNECT ACQUIRER IN PROCESSING TRANSACTIONS. NEITHER VISA NOR CYBERSOURCE WILL BE RESPONSIBLE OR LIABLE FOR RESELLER/PARTNER BOARDING MERCHANTS OR ENABLING MERCHANT PROCESSING IN VIOLATION OF THE TERMS AND CONDITIONS IMPOSED BY THE RELEVANT VISA PLATFORM CONNECT ACQUIRER.

TERMS OF USE APPLICABLE TO CARD NETWORK TOKENS

The following terms and conditions govern your use, receipt and/or possession of Card Network Tokens.

1. **DEFINTIONS.** Capitalized terms used herein shall have the following meanings:

    a. "**Card Network PAN**" means a number that is associated with a Payment Network for purposes of card transactions, all in accordance with Payment Network Rules.

    b. "**Card Network Token**" means a number provided by Cybersource pursuant to your use of Token Management Service ("TMS") that (i) is mapped to and is a surrogate for a Card Network PAN; and (ii) to use the underlying Card Network PAN number in accordance with the Cybersource Documentation.

    c. "**Payment Network Rules**" means the operating rules, bylaws, schedules, supplements and addenda, manuals, instructions, releases, specifications and other requirements, as may be amended from time to time, of any of the Payment Networks.

    d. "**Payment Network(s)"** means Visa, MasterCard, American Express, Discover Financial Services, and any affiliates thereof or any other payment network applicable to these Terms.

2. **LIMITATIONS ON USE OF CARD NETWORK TOKENS.** You agree to the following with respect to your use, receipt and/or possession of Card Network Tokens:

    a. You shall not maintain or create a mapping of the Card Network Token to the associated Card Network PAN.

    b. Upon request by Cybersource and/or the applicable Payment Network, you shall use commercially reasonable efforts to delete any or all of the Card Network Tokens. You acknowledge and agree that Cybersource or the applicable Payment Network may request that you delete any Card Network Token at their sole discretion.

    c. You shall not initiate any transaction with a Card Network Token without appropriate consent from and disclosures to the cardholder, including any necessary consents in order for the applicable Payment Network to receive, store, process and share any data in order to deliver the token service. Except as authorized in accordance with the applicable Payment Network Rules, you must use the Card Network Token only for transactions that are authorized, cleared and settled through the applicable Payment Network.

    d. You shall not use a Card Network Token in a manner that a Card Network PAN cannot be used under the applicable Payment Network Rules. You agree that your responsibility for use of Card Network Tokens is the same as your responsibilities for use of Account Numbers under the applicable Payment Network Rules.

e. You agree that the Payment Network Rules govern your relationship with the applicable Payment Network and use of Card Network Tokens as if the Card Network Tokens were Card Network PANs. You must comply with all applicable Payment Network Rules, as determined by the applicable Payment Network.

f. You agree that any Card Network Tokens will be stored in compliance with PCI-DSS and such storage is subject to your representations and warranties set forth in the applicable agreement between you and Cybersource.

g. If you are a Reseller or Partner, to enable American Express Network Tokens, you must have a direct acquiring or processing agreement signed with American Express in order to support American Express Network Tokens on behalf of your merchants.

3. **CARD ART.** Cybersource may pass through rights allowing you to use, reproduce, display and provide issuers' trademarks and issuer-provided card art (collectively, "Issuer IP") on a non-exclusive basis in strict accordance with the meta-data made available to you and such issuers' branding guidelines (which may be updated by issuer from time to time), for use and display solely for use with Card Network Tokens provisioned via TMS. You agree that you will not and will not cause your affiliates or agents to alter the meta-data in any way.

Introduction to the Token Management Service

The Token Management Service (TMS) enables you to replace personally identifiable information (PII), such as the primary account numbers (PANs), with unique tokens. These tokens do not include the PII data, but act as a placeholder for the personal information that would otherwise need to be shared. By using tokens, businesses can provide a secure payment experience, reduce the risk of fraud, and comply with industry consumer security regulations such as PCI-DSS.

TMS links tokens across service providers, payment types, and channels for sellers, acquirers, and technology partners. TMS tokenizes, securely stores, and manages the primary account number (PAN), the payment card expiration date, electronic check details, and customer data. TMS also enables you to create a network token of a customer's payment card.

Important: Due to mandates from the Reserve Bank of India, Indian merchants cannot store PANs. Use network tokenization instead.

You can manage sensitive data securely by creating, retrieving, updating, and deleting tokens through the TMS API.

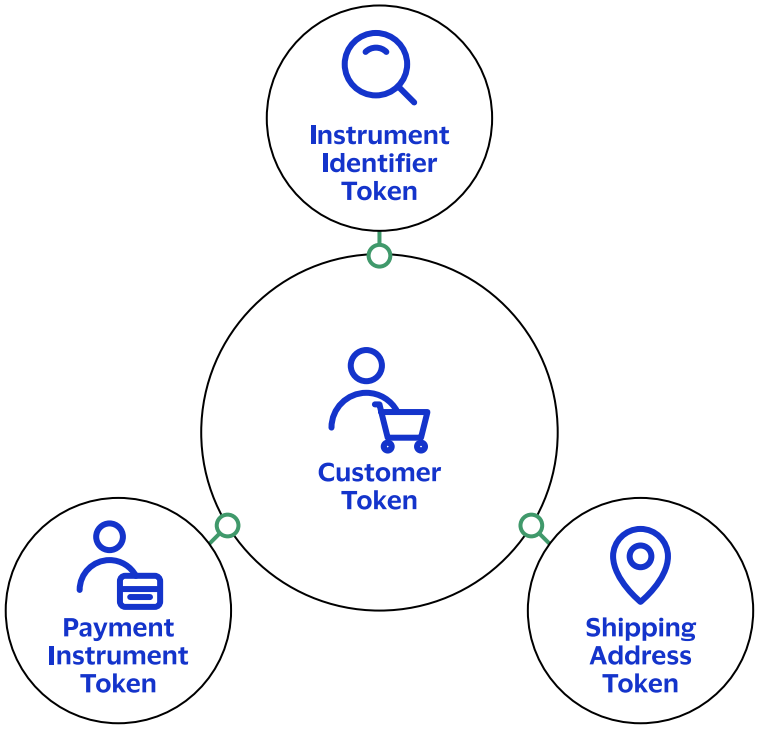
TMS simplifies your PCI DSS compliance. TMS passes tokens back to you that represent this data. You then store these tokens in your environment and databases instead of storing customer payment details.

TMS protects sensitive payment information through tokenization and secures and manages customer data using these token types:

- Customer tokens
- Instrument identifier tokens
- Payment instrument tokens
- Shipping address tokens

These TMS tokens can be used individually, or they can be associated with one customer token:

TMS Token Types

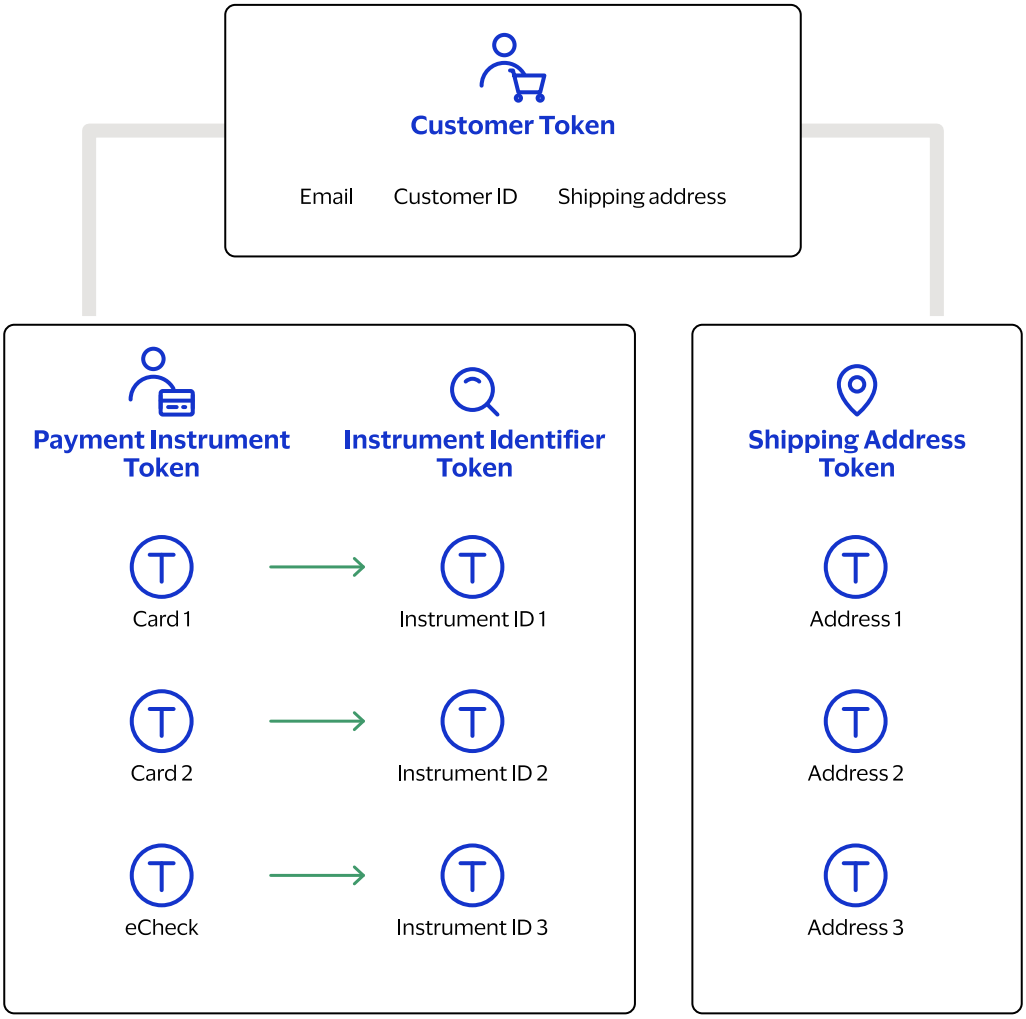


Types of Tokens

These tokens comprise the types of TMS tokens:

- **Customer Token**: Contains customer's email address, customer ID, shipping address (stored in a token), and other related data.
- **Shipping Address Token**: Contains the shipping address associated with a customer token.
- **Payment Instrument Token**: Contains the complete billing details for the payment type including cardholder name, expiration date, and billing address.
- **Instrument Identifier Token**: Contains the tokenized primary account number (PAN) for card payments as well as the associated network token or US or Canadian bank account number and routing number.
- **Network Token**: Network tokens pass through an acquirer and are de-tokenized by the payment network or issuer. For customer-initiated transactions, they require a cryptogram. Network tokens are mapped to instrument identifier tokens.

TMS Token Types



Customer Tokens

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

Related information

Manage Customer Tokens *(on page 91)*
Payments with Customer Tokens *(on page 109)*

Shipping Address Tokens

The shipping address token contains the shipping address information associated with a customer token. This token includes any shipping address details, including the recipient's first and last name, company, shipping address, email, and phone number. A customer can have one or more shipping addresses, with one allocated as the customer's default shipping address.

Related information

Manage Shipping Address Tokens *(on page 126)*
Payments with Shipping Address Tokens *(on page 145)*

Payment Instrument Tokens

The payment instrument token contains the complete billing details for the payment type including cardholder name, expiration date, and billing address. These are standalone payment instruments that cannot be associated with a customer.

Related information

Manage Payment Instrument Tokens *(on page 206)*
Payments with Payment Instrument Tokens *(on page 233)*

Instrument Identifier Tokens

Instrument identifier tokens represent tokenized payment account numbers. Tokenized payment account information includes a primary account number (PAN) for card payments, or a US or Canadian bank account number and routing number for an ACH bank account. An instrument identifier token can exist independently, or it can be associated with a payment instrument.

An instrument identifier token can also contain an associated network token.

Instrument identifier tokens are associated with these features:

Card Art

TMS card art helps your customers select a card. See Card Art (on page 344).

Enrollable Network Tokens

TMS can enroll certain *network tokens* in an instrument identifier token to be used for future payments. Future payments require only the instrument identifier token for the payment information. The types of network tokens you can enroll into an instrument identifier are tokens used for in-app payment methods such as:

- Android Pay
- Apple Pay
- Chase Pay
- Google Pay
- Samsung Pay
- Visa Click to Pay

See Create an Instrument Identifier for Enrollable Network Tokens (on page 248).

Push Provisioning

Push provisioning connects you with participating issuers to quickly provide credentials to your customers. See Push Provisioning for Network Tokens (on page 339).

Related information

Manage Instrument Identifier Tokens *(on page 243)*
Payments with Instrument Identifier Tokens *(on page 271)*

Network Tokens

When a TMS token is used in a transaction, the TMS token is de-tokenized, and the PAN is sent to the issuer for authorization. The primary account number (PAN) is still exchanged as the transaction is processed. However, the PAN is removed from transaction processing and replaced with network tokens, making the transaction more secure.

The network scheme generates network tokens. A token replaces customer card information in order to ensure secure transactions. Network tokens are mapped to instrument identifier tokens. The minimum card data required in order to request a network token is the PAN and the expiration date.

Using a network token has benefits:

- Improved authorization rates for credentials-on-file (COF) and recurring payments.
- Real-time card information updates with life-cycle management. See Network Token Life-Cycle Management (on page 72) for more information. When the customer's card details change, you can receive the updated information automatically. See Manage Webhook Subscriptions (on page 78) for more information on managing webhooks.
- Improved customer tracking through the payment account reference (PAR), which is a consumer identifier that is less sensitive than the PAN. The PAR can be exchanged as the transaction is processed.

Network tokens can be provided for merchants and partners.

Important: American Express does not support the payment facilitator (PayFac) model for processing network tokens. Contact your American Express representative for more information.

Related information

Payments with Network Tokens for Merchants *(on page 297)*
Manage Network Tokens for Partners *(on page 305)*
Manage Merchant-Initiated Transactions and Network Tokens *(on page 301)*

Token Management Service Workflows

Tokenization workflows:

- PAN Tokenization Process Using TMS (on page 26)
- Network Token Tokenization Process (on page 27)

Network tokens workflows—merchant model:

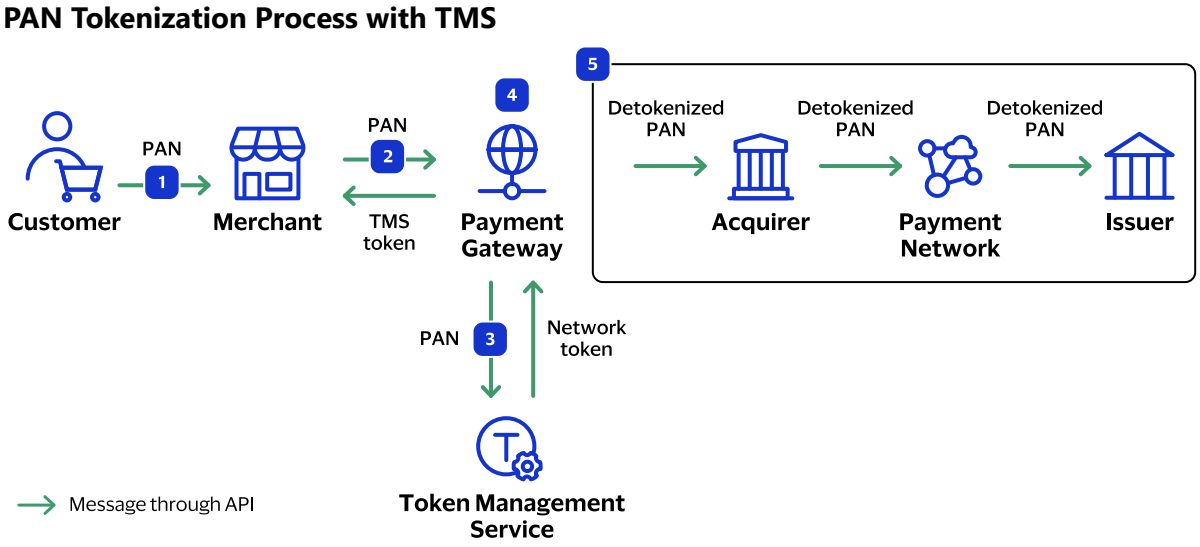
- Network Token Provisioning—Merchant Model (on page 30)
- Network Token Authorizations (CIT)—Merchant Model (on page 33)
- Network Token Authorizations (MIT)—Merchant Model (on page 35)

Network tokens workflows—partner model:

- Network Token Onboarding—Partner Model (on page 69)
- Network Token Provisioning—Partner Model (on page 37)
- Network Token Authorizations (CIT)—Partner Model (on page 40)
- Network Token Authorizations (MIT)—Partner Model (on page 42)

PAN Tokenization Process Using TMS

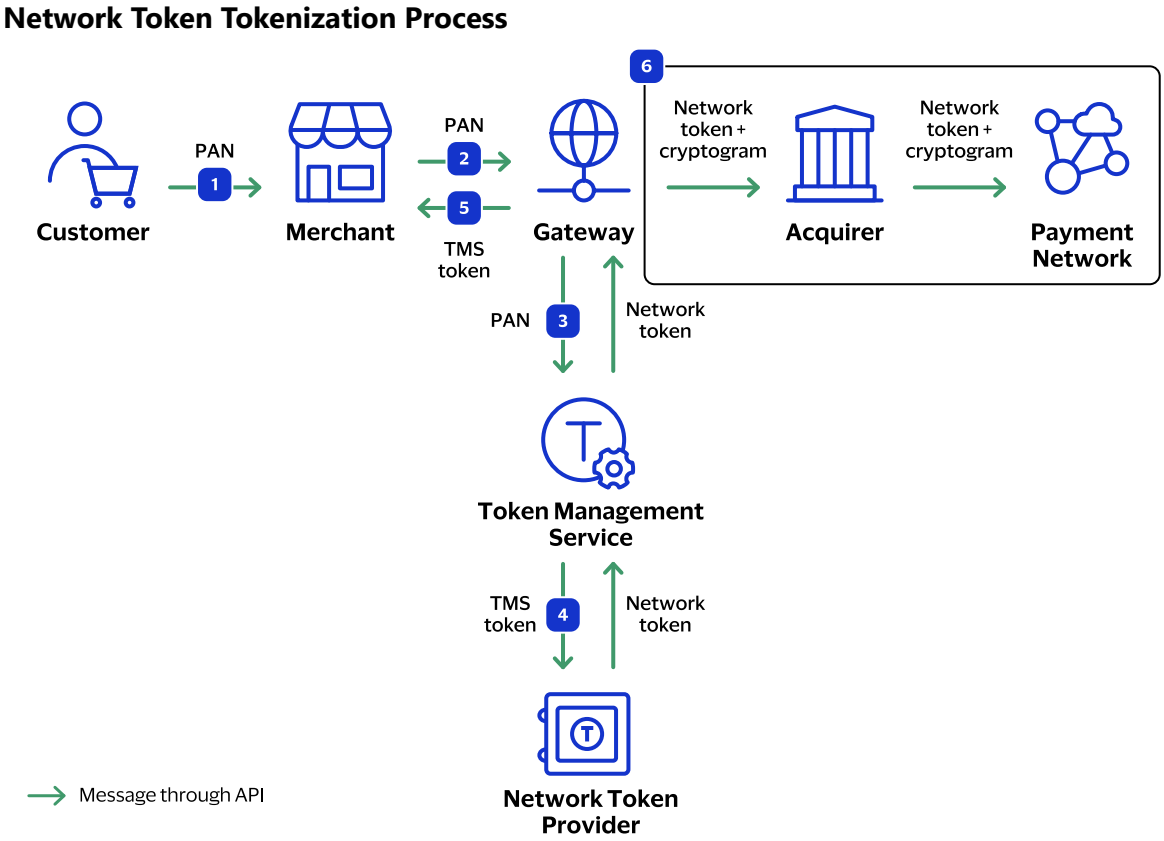
This workflow shows the tokenization process for TMS tokens.



1. The customer makes a purchase on the merchant's website using a PAN.
2. The merchant sends the PAN to the Cybersource gateway.
3. TMS creates a token for the merchant to store.
4. Cybersource detokenizes the TMS token when it is used in a transaction.
5. The detokenized PAN is exchanged across the payment ecosystem.

Network Token Tokenization Process

This workflow shows the tokenization process for network tokens.

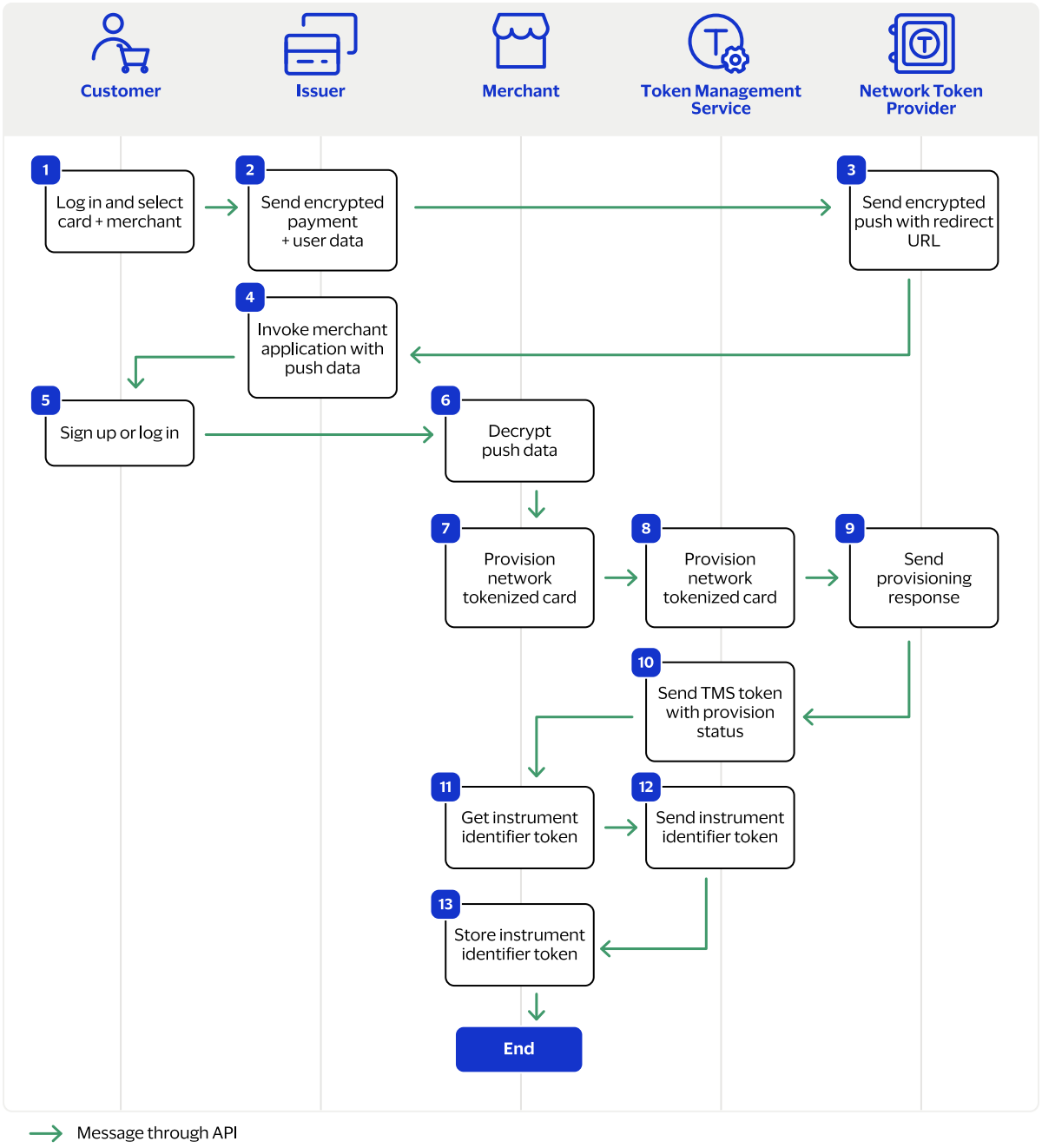


1. The customer makes a purchase on the merchant's website using a PAN.
2. The merchant sends the PAN to the gateway.
3. TMS creates a token.
4. TMS provisions a network token and links it to the TMS token.
5. The merchant stores the TMS token for subsequent transactions.
6. The network token and cryptogram are exchanged throughout the payment ecosystem.

Push Provisioning Process

This workflow shows the process for push provisioning.

Push Provisioning Process



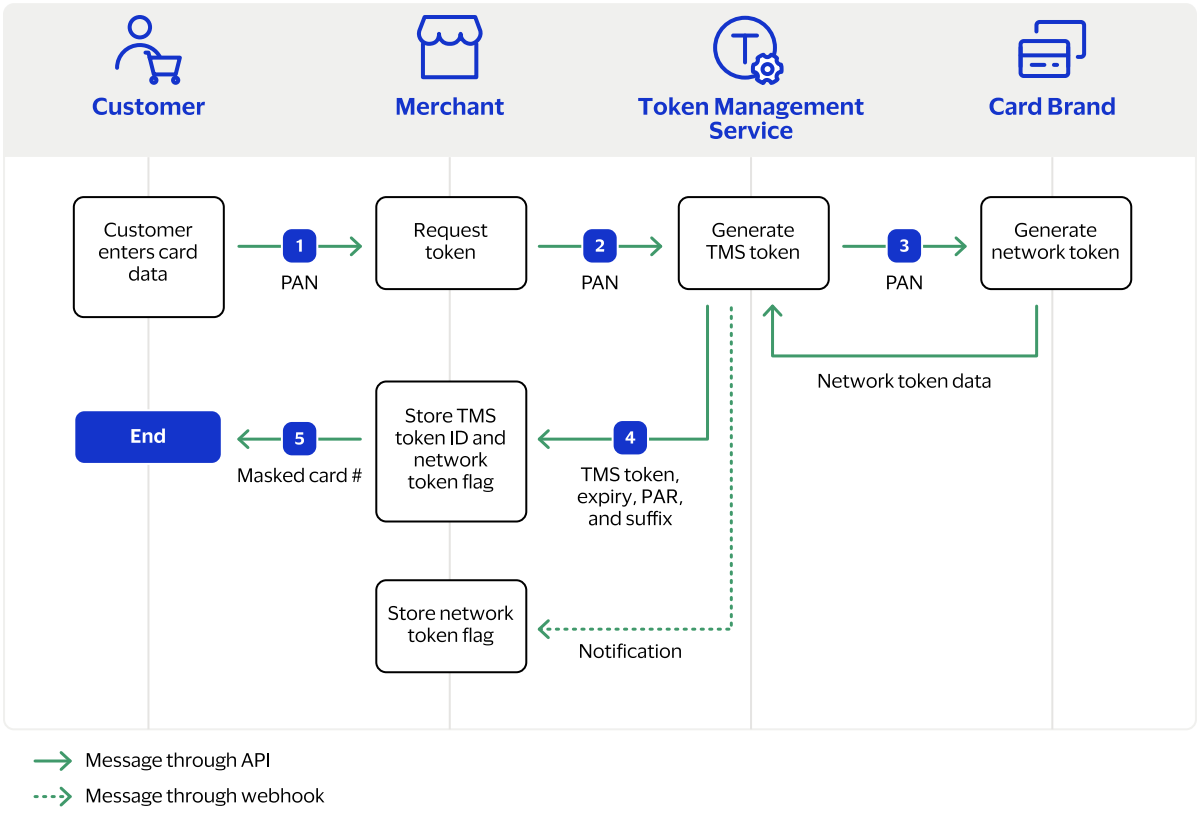
1. The customer logs in to their bank account and chooses a card and merchant.
2. The issuer sends the encrypted payment and user data to the network token provider.
3. The network token provider sends the encrypted payment and user data to the issuer.
4. The issuer invokes the merchant application with the token request push data.
5. The customer registers for a merchant account or logs into an existing account.

6. You decrypt the push data.
7. You send a request to TMS to provision a network tokenized card.
8. TMS sends a request to the network token provider to provision the tokenized card.
9. The network token provider sends the provisioning response to TMS.
10. TMS sends you the TMS token along with the provisioning status.
11. Using the response sent from TMS, you send a request to TMS to retrieve the instrument identifier token. See Retrieve an Instrument Identifier (on page 253).
12. TMS sends you the instrument identifier token.
13. You store the instrument identifier token for future transactions. Your designation as a merchant or a partner determines how you use an instrument identifier token in an authorization. For more information, see:
    - Merchant: Authorize a Payment with an Instrument Identifier (on page 277)
    - Partner: Retrieve Network Token Payment Credentials (on page 323)

Network Token Provisioning for Merchants

This workflow shows the process of network token provisioning for merchants.

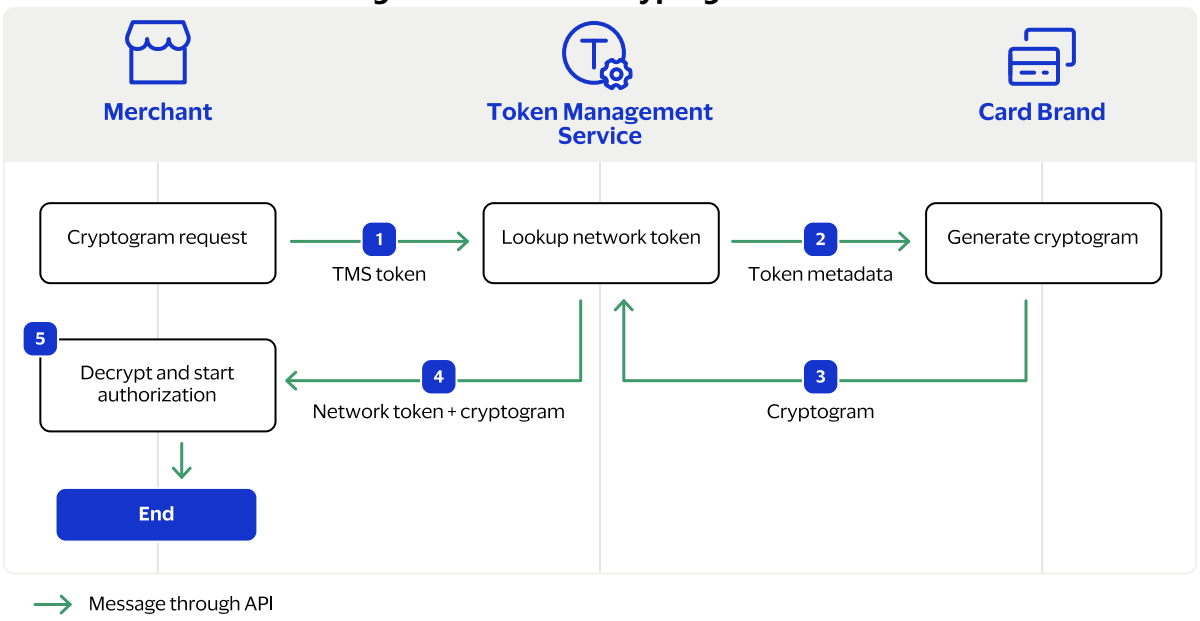
Network Token Provisioning for Merchants: Tokenizing PAN



Tokenizing PAN

1. The customer enters their card data and sends the PAN to the merchant.
2. The merchant sends the PAN to Token Management Service.
3. Token Management Service generates a TMS token and synchronously provisions a network token from the card brand.
4. Token Management Service sends the merchant the TMS token, expiration date, suffix, and payment account reference (PAR).
5. The merchant stores the TMS token ID and network token flag and sends the customer the masked card number.

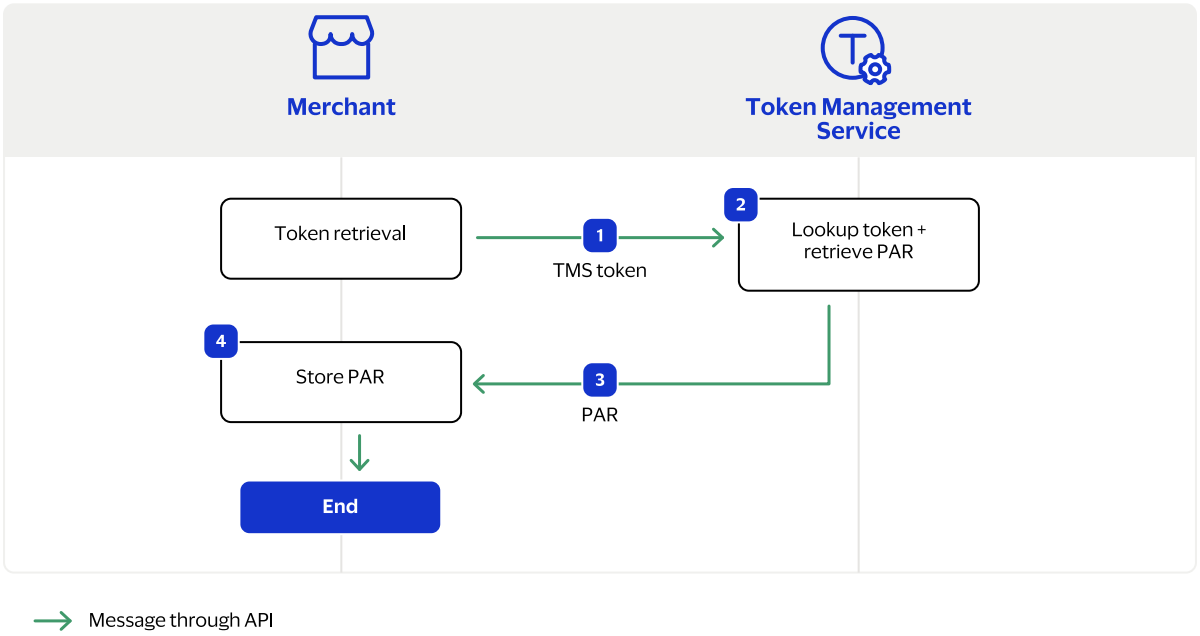
Network Token Provisioning for Merchants: Cryptogram Retrieval



Cryptogram Retrieval

1. The merchant requests the cryptogram using a TMS token from Token Management Service.
2. Token Management Service looks up the network token and sends the token metadata to the card brand.
3. The card brand generates the cryptogram and sends it to Token Management Service.
4. Token Management Service sends the network token and cryptogram to the merchant.
5. The merchant uses the network token along with the cryptogram to start the authorization.

Network Token Provisioning for Merchants: PAR Retrieval



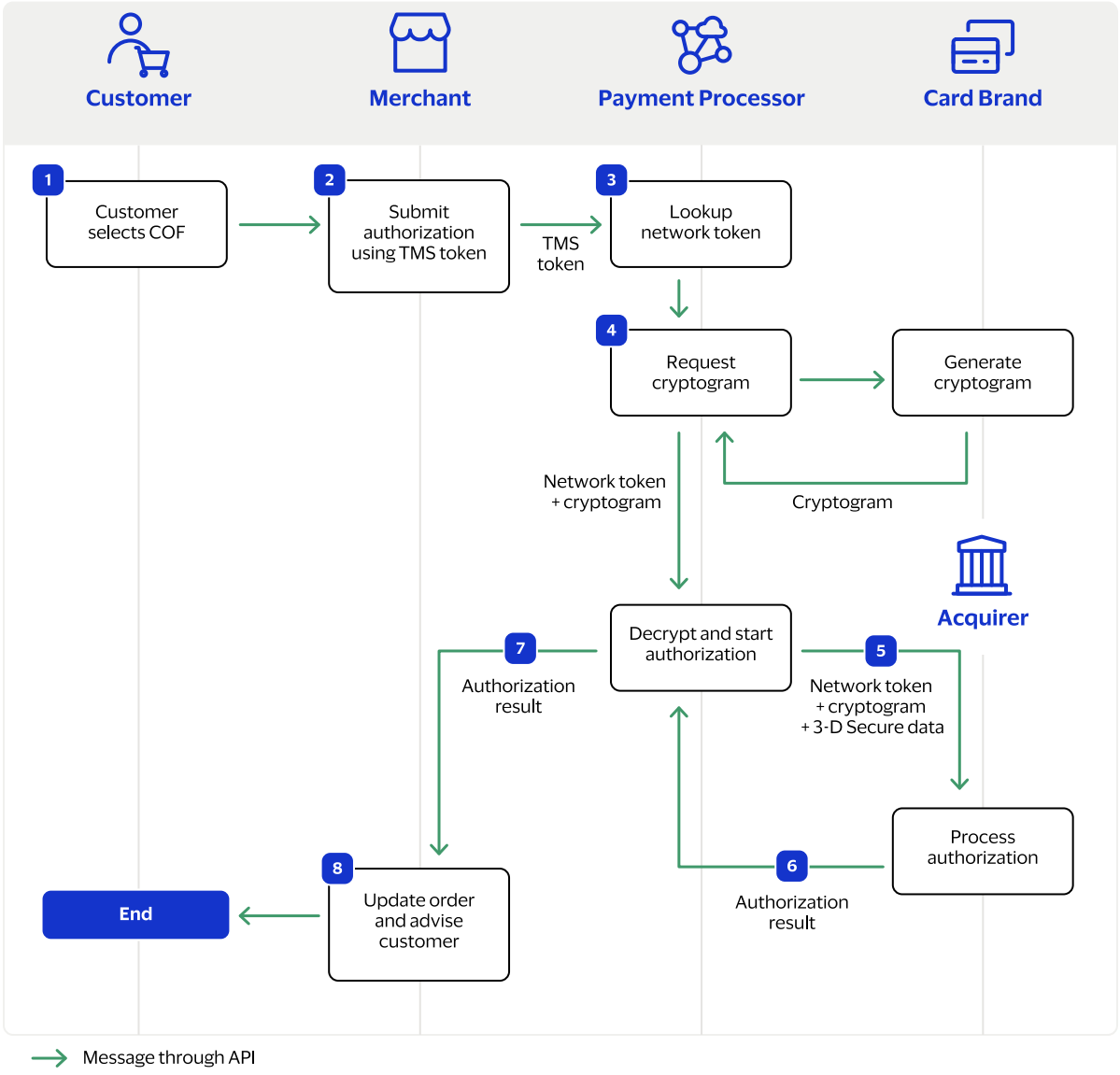
PAR Retrieval

1. The merchant retrieves the TMS token and sends to Token Management Service.
2. Token Management Service looks up the token and retrieves the PAR.
3. Token Management Service sends the PAR to the merchant.
4. The merchant stores the PAR.

Network Token CIT for Merchants

This workflow shows a credentials-on-file (COF) authorization using a network token for a customer-initiated transaction (CIT).

Network Token CIT Authorizations for Merchants



1. The customer makes a purchase and selects COF.
2. The merchant submits an authorization to the payment processor using a TMS token.
3. The payment processor uses the TMS token to look up the network token.
4. The payment processor requests the cryptogram generated by the card brand.
5. The payment processor sends the network token, cryptogram, and 3-D Secure data to the acquirer in the authorization request.

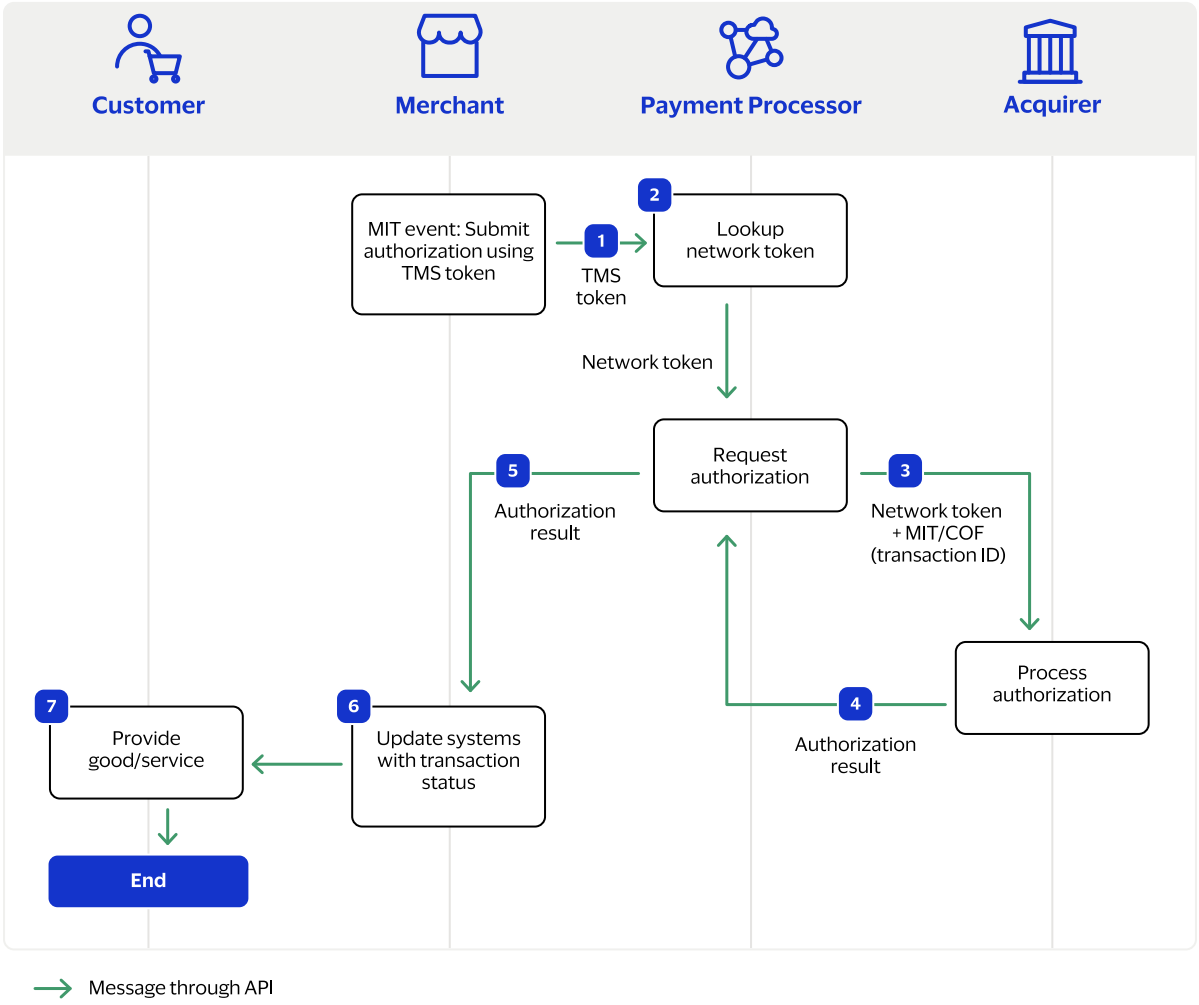
6. The acquirer processes the authorization and sends the authorization result to the payment processor.
7. The payment processor sends the authorization result to the merchant.
8. The merchant updates the order and advises the customer on how to proceed depending on the authorization result.

Network Token MIT for Merchants

This workflow shows a credentials-on-file (COF) authorization using a network token for a merchant-initiated transaction (MIT).

Important: Before you can process a MIT, the customer must have previously made a purchase and given consent for you to store their payment credentials.

Network Token MIT Authorizations for Merchants



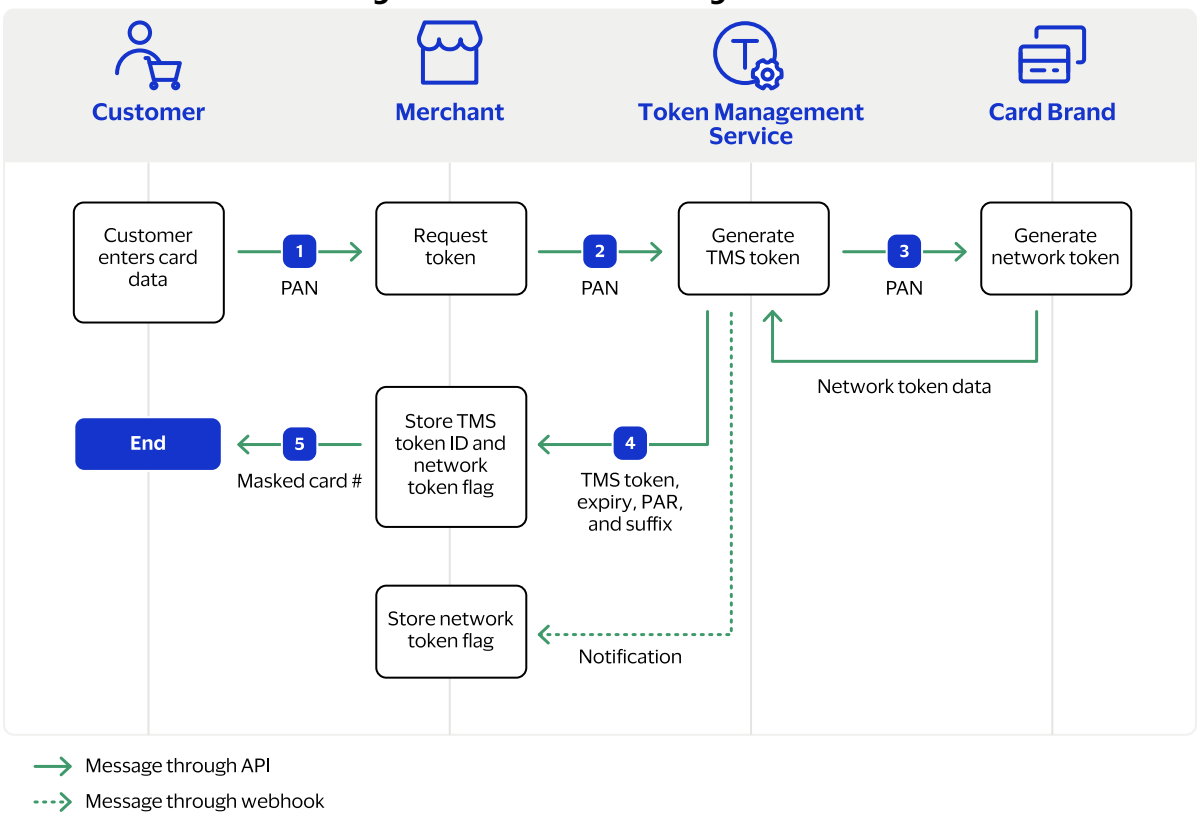
1. The merchant sends the TMS token to the payment processor.
2. The payment processor looks up the network token associated with the TMS token.
3. The payment processor sends the network token and MIT COF data to the acquirer in the authorization request.
4. The acquirer processes the authorization and sends the authorization result to the payment processor.

5. The payment processor sends the authorization result to the merchant.
6. The merchant updates the system to reflect the status of the transaction.
7. The customer provides goods/service.

Network Token Provisioning for Partners

This workflow shows the process of network token provisioning for partners.

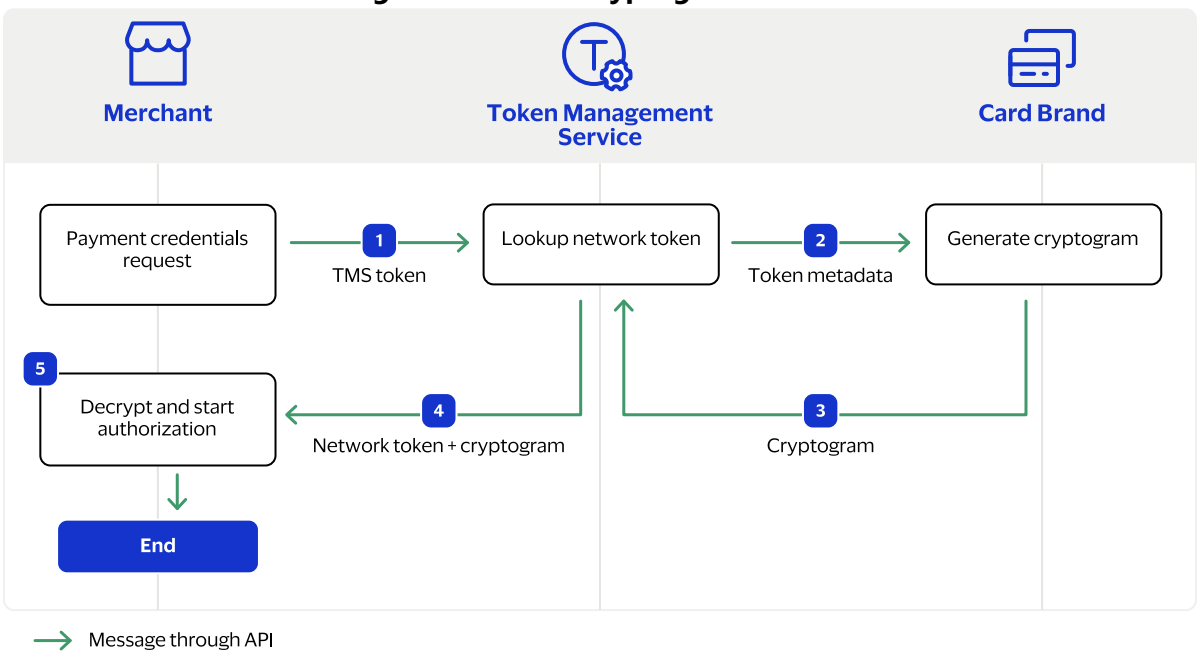
Network Token Provisioning for Partners: Tokenizing PAN



Tokenizing PAN

1. The customer enters their card data and sends the PAN to the merchant.
2. The merchant sends the PAN to Token Management Service.
3. Token Management Service generates a TMS token and synchronously provisions a network token from the card brand.
4. Token Management Service sends the merchant the TMS token, expiration date, suffix, and payment account reference (PAR).
5. The merchant stores the TMS token ID and network token flag and sends the customer the masked card number.

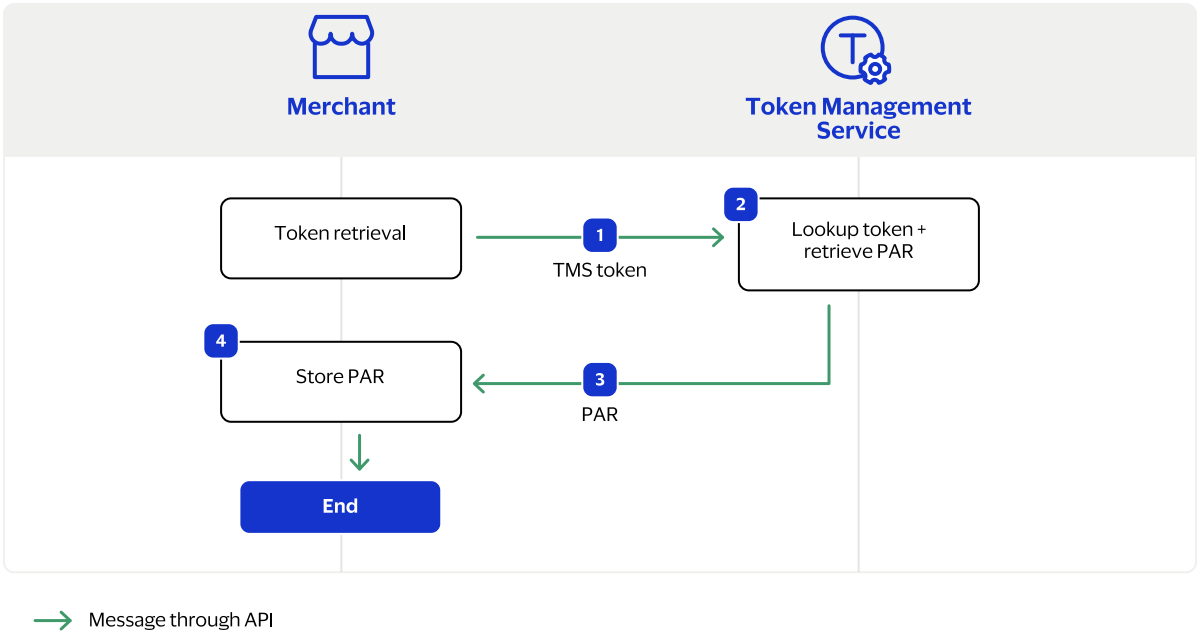
Network Token Provisioning for Partners: Cryptogram Retrieval



Cryptogram Retrieval

1. The merchant requests the payment credentials using a TMS token from Token Management Service.
2. Token Management Servicelooks up the network token and sends the token metadata to the card brand.
3. The card brand generates the cryptogram and sends it to Token Management Service.
4. Token Management Service sends the network token and cryptogram to the merchant.
5. The merchant uses the network token along with the cryptogram to start the authorization.

Network Token Provisioning for Partners: PAR Retrieval



PAR Retrieval

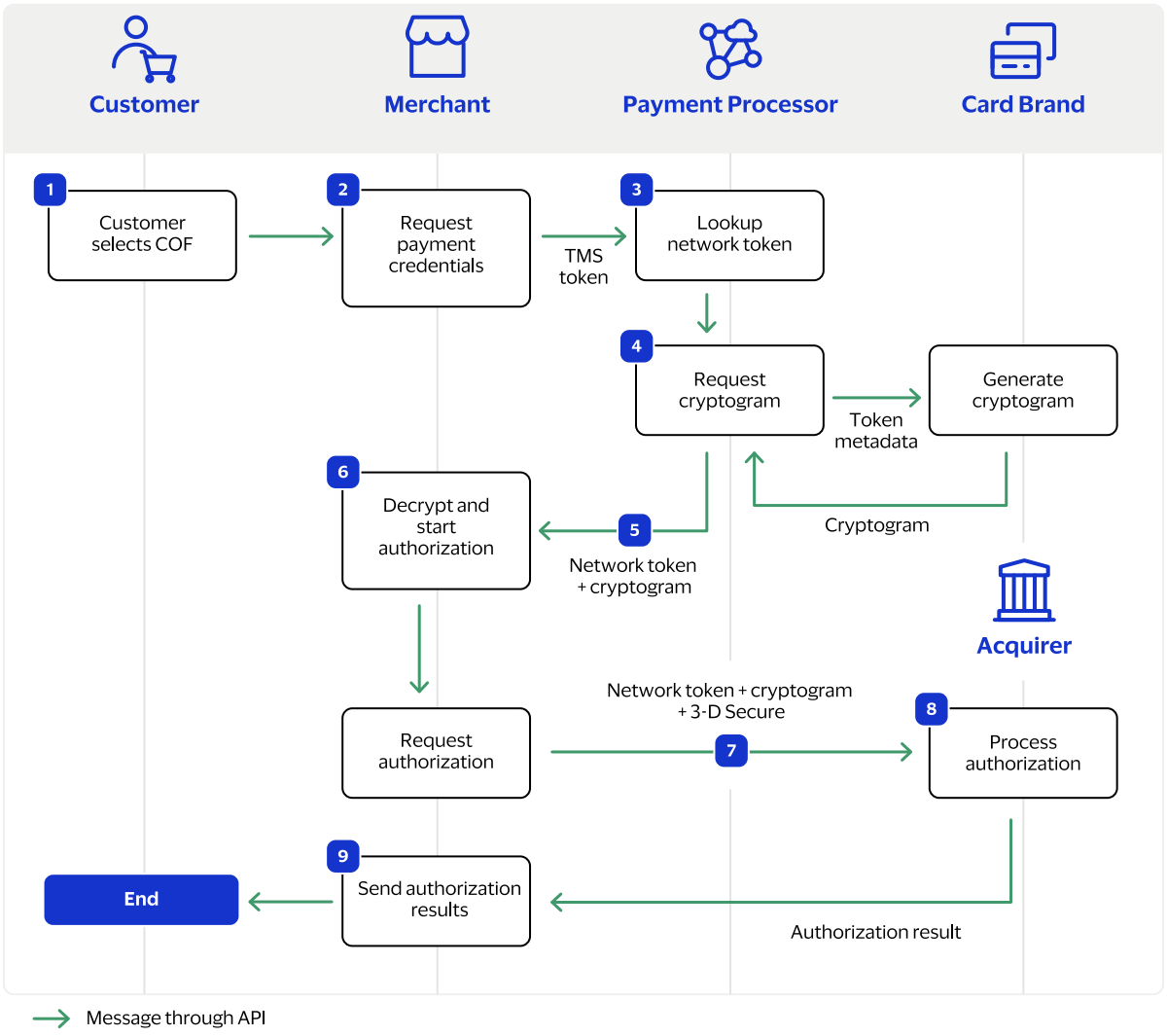
1. The merchant retrieves TMS token and sends it to Token Management Service.
2. Token Management Service looks up the token and retrieves the PAR.
3. Token Management Service sends the PAR to the merchant.
4. The merchant stores the PAR.

Network Token CIT for Partners

This workflow shows a credentials-on-file (COF) authorization using a network token for a customer-initiated transaction (CIT).

The workflow begins when the customer makes a purchase from the merchant and selects a COF during payment.

Network Token CIT Authorizations for Partners



1. The customer makes a purchase and selects COF.
2. The merchant requests the payment credentials and sends the TMS token to the payment processor.
3. The payment processor uses the TMS token to look up the network token.
4. The payment processor requests the cryptogram generated by the card brand.
5. The payment processor sends the network token and cryptogram to the merchant.
6. The merchant uses the network token along with the cryptogram to start the authorization.

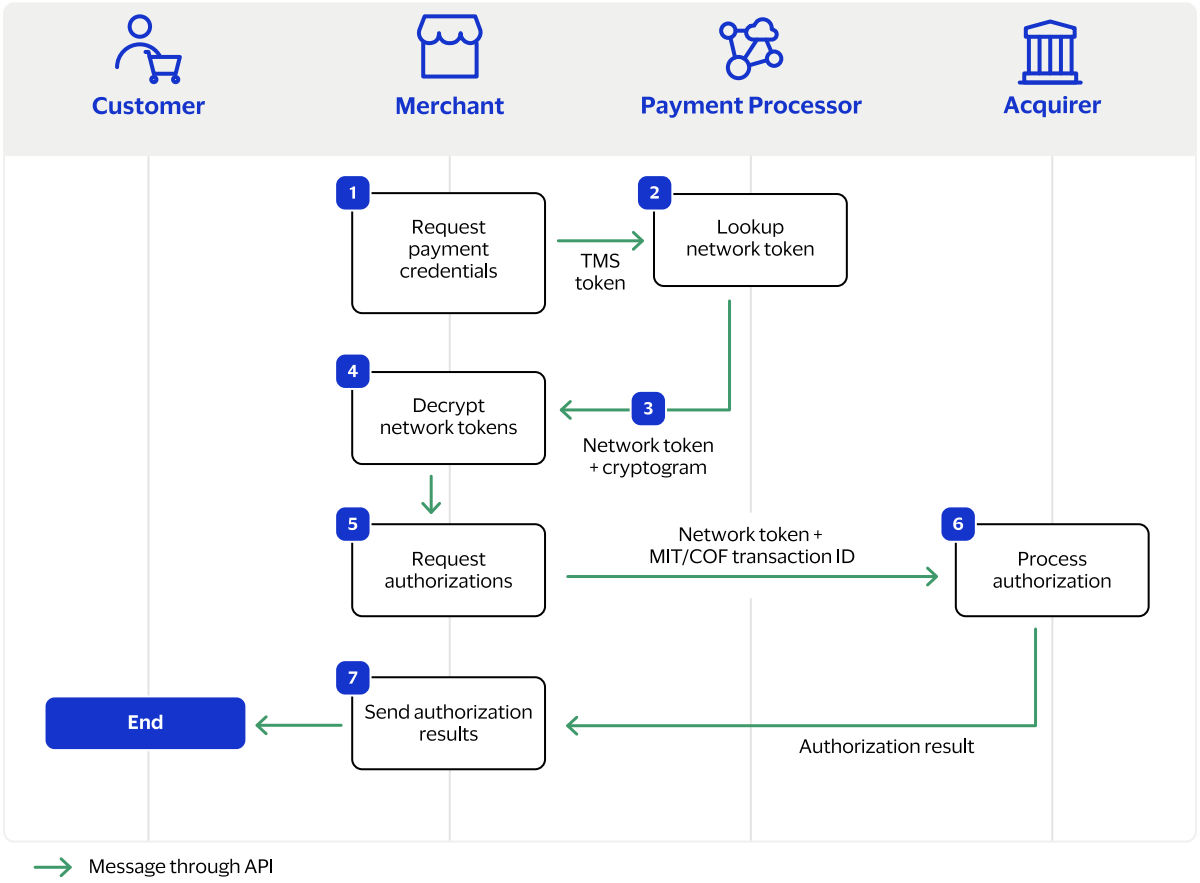
7. The merchant sends the network token, cryptogram, and 3-D Secure data to the acquirer in the authorization request.
8. The acquirer processes the authorization and sends the authorization result to the merchant.
9. The merchant sends the customer the authorization result from the acquirer.

Network Token MIT for Partners

This workflow shows a credentials-on-file (COF) authorization using a network token for a merchant-initiated transaction (MIT).

Important: Before you can process a MIT, the customer must have previously made a purchase and given consent for you to store their payment credentials.

Network Token MIT Authorizations for Partners



1. The merchant requests the payment credentials and sends the TMS token to the payment processor.
2. The payment processor uses the TMS token to look up the network token.
3. The payment processor sends the network token and cryptogram to the merchant.
4. The merchant uses the network token along with the cryptogram to start the authorization.

5. The merchant sends the network token and MIT COF data to the acquirer in the authorization request.
6. The acquirer processes the authorization and sends the authorization result to the merchant.
7. The merchant sends the customer the authorization result from the acquirer.

Requesting the Token Management Service API

Before requesting the Token Management Service (TMS) API, you must already have a Business Center account. If you do not, you can create an evaluation account.

Follow these steps to request the TMS API:

1. Authenticate to the API using either HTTP signature authentication or JSON Web Token (JWT) authentication.

    a. A Base64-encoded shared secret key is passed in the headers you generate for HTTP signature authentication.

    See Shared Secret Key Pair in *Getting Started with the REST API* for instructions.

    b. A P12 Certificate is passed in the headers you generate for JWT authentication.

    See Create a P12 Certificate in the *Getting Started with the REST API* for instructions.

2. Specify one of the following hosts in the URL:

    a. **Sandbox:** `POST https://apitest.cybersource.com`

    b. **Production:** `POST https://api.cybersource.com`

    c. **Production in India:** `POST https://api.in.cybersource.com/`

3. Append the resource, such as, `/tms/v2/customer` to the host URL. For example, `https://api.cybersource.com/tms/v2/customer`.

4. Pass your request using a `HTTP` `GET`, `POST`, `PATCH` or `DELETE` method as specified in each API operation.

HTTP Response Headers

Response Header	Possible Values	Description
instrumentidentifier-created	`true` or `false`	This value indicates whether a new instrument identifier was created. For example, you have never tokenized this PAN or bank account, or an existing one was returned.

Related information

Getting Started with the REST API—Secure Communication Requirements

Case Sensitivity

Token IDs are not case sensitive. The following requests return the same resource:

```
GET /instrumentidentifiers/49C26351BF7D8765E05333B9d30AA9DB
```

```
GET /instrumentidentifiers/49c26351bf7d8765e05333b9d30aa9db
```

Important: Unlike the token ID in the request URL, all request fields are case sensitive.

List matching rules:

- Accept any case (web, WEB, WeB).
- Store the expected case (WEB).
- Return the expected case (WEB) metadata.

Metadata

Token type structures such as instrument identifiers and payment instruments contain a metadata map that contains data about the creator.

A metadata map is returned for every token type in a response to an HTTP POST, GET, and PATCH request.

Example: Metadata from a Response

```
"metadata": {
"creator": "mid1"
}
```

Patching Considerations

Patching within TMS is based on JSON Merge Patch (RFC7396), in which changes follow the same structure being modified as that of a POST request, rather than JavaScript Object Notation (JSON) Patch (RFC6902), in which changes are expressed as a set of actions.

A PATCH request is different from a PUT request in that only the fields that must be changed need to be provided in the request, and those changes are merged with the existing record.

Here are some rules to consider:

- When a field is to be removed, you can remove a field by entering a value of `null`.
- When a field is set to `null`, and it does not exist in the current record, it is ignored.
- You can remove groups of fields by setting the parent container to `null`.

Important: Array values are patched as a whole, so in the patch request, provide the final value that is expected after the patch.

Patching Examples

Below are some use-case examples of patching rules.

Example: Updating Expiration Month and Year Values

You can get the existing values by sending a GET request to the payment instrument ID as shown below:

```
GET /tms/v1/paymentinstrument/<id>
```

The response is shown below:

```
{
  "_links": {
    "self": {
      "href": "https://api.cybersource.com/tms/v1/paymentinstruments/9000000000000
000002001"
     }
  },
  "id": "900000000000000002001",
  "object": "paymentInstrument",
  "state": "ACTIVE",
```

```
  "card": {
    "expirationMonth": "09",
    "expirationYear": "2017",
    "type": "visa",
    "issueNumber": "01"
  },
  "_embedded": {
    "instrumentIdentifier": {
      "_links": {
        "self": {
          "href": "https://api.cybersource.com/tms/v1/instrumentidentifiers/900000
0000000000001001"
        }
      },
      "id": "900000000000000001001",
      "object": "instrumentIdentifier",
      "state": "ACTIVE",
      "card": {
        "number": "411111XXXX11112"
      }
    }
  }
}
```

To update just the **card.expirationMonth** and **card.expirationYear** fields, send the following PATCH request:

```
PATCH /tms/v1/paymentinstrument/<id>
{
  "card": {
    "expirationMonth": "10",
    "expirationYear": "2020"
  }
}
```

You can see the new values by issuing another GET request to `/tms/v1/paymentinstrument/<id>`. The response is shown below.

```
{
  "_links": {
    "self": {
      "href": "https://api.cybersource.com/tms/v1/paymentinstruments/9000000000000
000002001"
    }
  },
  "id": "900000000000000002001",
  "object": "paymentInstrument",
```

```
  "state": "ACTIVE",
  "card": {
    "expirationMonth": "10",
    "expirationYear": "2020",
    "type": "visa",
    "issueNumber": "01"
  },  "_embedded": {
    "instrumentIdentifier": {
      "_links": {
        "self": {
          "href": "https://api.cybersource.com/tms/v1/instrumentidentifiers/900000
0000000000001001"
        }
      },
      "id": "900000000000000001001",
      "object": "instrumentIdentifier",
      "state": "ACTIVE",
      "card": {
        "number": "411111XXXX11112"
      }
    }
  }
}
```

Example: Removing Card Issue Number (Single Field) and Buyer Information (Container)

First, send a GET request to `/tms/v1/paymentinstrument/<id>` to see the current values:

```
{
  "_links": {
    "self": {
      "href": "https://api.cybersource.com/tms/v1/paymentinstruments/9000000000000
000002001"
    }
  },
  "id": "900000000000000002001",
  "object": "paymentInstrument",
  "state": "ACTIVE",
  "card": {
    "expirationMonth": "09",
    "expirationYear": "2017",
    "type": "visa",
    "issueNumber": "01"
  },
  "buyerInformation": {
    "companyTaxID": "12345",
    "currency": "USD"
  },
```

```
  "_embedded": {
    "instrumentIdentifier": {
      "_links": {
        "self": {
          "href": "https://api.cybersource.com/tms/v1/instrumentidentifiers/900000
0000000000001001"
        }
      },
      "id": "900000000000000001001",
      "object": "instrumentIdentifier",
      "state": "ACTIVE",
      "card": {
        "number": "411111XXXX11112"
      }
    }
  }
}
```

Then send a PATCH request to `/tms/v1/paymentinstrument/<id>` and include the following payload:

```
{
  "card": {
    "issueNumber": null
  },
"buyerInformation": null
}
```

The result can be seen in the next GET request to `/tms/v1/paymentinstrument/<id>`:

```
{
  "_links": {
    "self": {
      "href": "https://api.cybersource.com/tms/v1/paymentinstruments/9000000000000
000002001"
    }
  },
  "id": "900000000000000002001",
  "object": "paymentInstrument",
  "state": "ACTIVE",
  "card": {
    "expirationMonth": "09",
    "expirationYear": "2017",
    "type": "visa"
  }
  "_embedded": {
    "instrumentIdentifier": {
      "_links": {
```

```
        "self": {
          "href": "https://api.cybersource.com/tms/v1/instrumentidentifiers/900000
0000000000001001"
        }
      },
      "id": "900000000000000001001",
      "object": "instrumentIdentifier",
      "state": "ACTIVE",
      "card": {
        "number": "411111XXXX11112"
      }
    }
  }
}
```

Example: Patching an Array

Original value:

```
{
  "a": [
    {
      "b": "c",
      "d": "e"
    }
  ]
}
```

Patch payload:

```
{
  "a": [
    {
      "z": "y"
    }
  ]
}
```

Final value:

```
{
  "a": [
      {
      "z": "y"
```

```
    }
  ]
}
```

Pagination

Responses can indicate pagination if you include the **limit** and **offset** fields in your request.

Parameter	Description

`limit`	Controls the maximum number of items that can be returned for a single request. The default is 20; the maximum is 100.
	
	If you set a limit greater than 100, the following error results:
	
	`Http Status: 400 Bad Request`
	`Body`
	`{`
	`  "errors": [`
	`    {`
	`      "type": "invalidParameters",`
	`      "message": "Invalid parameter values",`
	`      "details": [`
	`        {`
	`          "name": "limit"`
	`        }`
	`      ]`
	`    }`
	`  ]`
	`}`

`offset`	Controls the starting point within the collection of results. Defaults to `0`. Setting a zero offset retrieves the first item in the collection. For example, if you have a collection of 15 items to be retrieved from a resource, and you specify `limit=5` , you can retrieve the entire set of results in three successive requests by varying the offset value: `offset=0`, `offset=5`, and `offset=10`. An offset greater than the number of results does not return an embedded object.

Pagination Response Header

Header	Description
`X-total-count`	Returns total records count regardless of pagination.

Pagination Response Body Fields

Field	Description
`"object":"collection"`	Shows that the response is a collection of objects.
`"offset": 40`	The offset parameter used in the request.
`"limit": 20`	The limit parameter used in the request.
`"count": 20`	The number of objects returned.
`"total": 87`	The total number of objects.

Examples

Pagination Example 1

This example shows a request for objects 41 to 60.

Request

```
GET https://api.cybersource.com/tms/v1/instrumentidentifiers/5BAAD18F8091052CE0539
399D30AAB2F/paymentinstruments?offset=40&limit=20
```

For merchants in India, the Production endpoint is `https://api.in.cybersource.com/`

Important:

- If you are on the first collection, the previous link would not be included.
- If you are on the last collection, the next link would not be included.
- All other links are always included. For example, if there was only one collection of results, the URL for `self`, `first`, and `last` links would be the same.

Response

```
{
  "_links": {
    "self": {
      "href":
 "https://api.cybersource.com/tms/v1/instrumentidentifiers/5BAAD18F8091052CE053939
9D30AAB2F/paymentinstruments?offset=40&limit=20"
    },
    "first": {
      "href":
 "https://api.cybersource.com/tms/v1/instrumentidentifiers/5BAAD18F8091052CE053939
9D30AAB2F/paymentinstruments?offset=0&limit=20"
    },
    "prev": {
      "href":
 "https://api.cybersource.com/tms/v1/instrumentidentifiers/5BAAD18F8091052CE053939
9D30AAB2F/paymentinstruments?offset=20&limit=20"
    },
    "next": {
      "href":
 "https://api.cybersource.com/tms/v1/instrumentidentifiers/5BAAD18F8091052CE053939
9D30AAB2F/paymentinstruments?offset=60&limit=20"
    },
    "last": {
      "href":
 "https://api.cybersource.com/tms/v1/instrumentidentifiers/5BAAD18F8091052CE053939
9D30AAB2F/paymentinstruments?offset=80&limit=20"
    }
  },
  "object":"collection",
  "offset": 40,
  "limit": 20,
  "count": 20,
  "total": 87,
  "_embedded": {
    <array data>
  }
}
```

Pagination Example 2 - Offset to Limit Relationship

This example shows a request for objects 3 to 6, from a total of 8 objects.

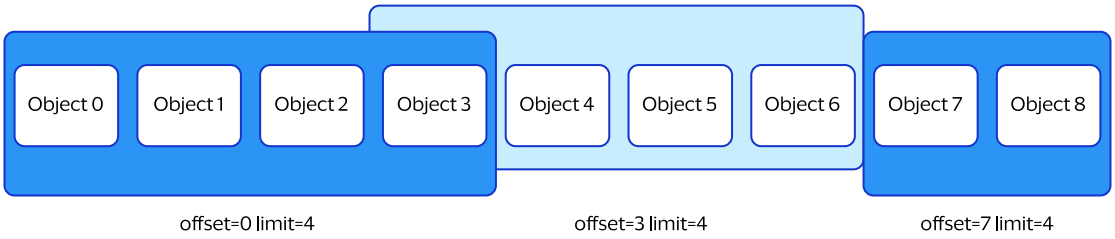
The example below shows the second collection of results and highlights that the previous page link will not change the user's original limit parameter value.

This means that the previous collection will contain objects 0-3, and therefore collection 1 and collection 2 will both contain object 3.

Request

```
GET https://api.cybersource.com/tms/v1/instrumentidentifiers/5BAAD18F8091052CE0539
399D30AAB2F/paymentinstruments?offset=3&limit=4
```

Offset to Limit Relationship



Response

```
{
  "_links": {
    "self": {
      "href":
 "https://api.cybersource.com/tms/v1/instrumentidentifiers/5BAAD18F8091052CE053939
9D30AAB2F/paymentinstruments?offset=3&limit=4"
    },
    "first": {
      "href":
 "https://api.cybersource.com/tms/v1/instrumentidentifiers/5BAAD18F8091052CE053939
9D30AAB2F/paymentinstruments?offset=0&limit=4"
    },
    "prev": {
      "href":
 "https://api.cybersource.com/tms/v1/instrumentidentifiers/5BAAD18F8091052CE053939
9D30AAB2F/paymentinstruments?offset=0&limit=4"
    },
    "next": {
      "href":
 "https://api.cybersource.com/tms/v1/instrumentidentifiers/5BAAD18F8091052CE053939
9D30AAB2F/paymentinstruments?offset=7&limit=4"
    },
    "last": {
      "href":
 "https://api.cybersource.com/tms/v1/instrumentidentifiers/5BAAD18F8091052CE053939
9D30AAB2F/paymentinstruments?offset=7&limit=4"
    }
  },
  "object":"collection",
  "offset": 3,
```

```
  "limit": 4,
  "count": 4,
  "total": 8,
  "_embedded": {
    <array data>
  }
}
```

Supported Processors

The processors listed below support customer and instrument identifier tokens, unless noted otherwise.

Processor	Payment Method
Visa Platform Connect	• Credit card—supports 0.00 pre-authorizations for Visa and Mastercard cards. • Credit card—supports 1.00 pre-authorizations for American Express, Discover, Diners Club, and JCB card types. • Debit card and prepaid card. • Payouts.

Test Card Numbers

Use these test card numbers to provision and test TMS tokens and network tokens.

All of the test card numbers listed here are enabled for card art. For more information on card art, see Card Art (on page 344).

Successful Network Token Provisioning

Use these test card numbers to provision network tokens. For Mastercard and Visa cards, replace the X in the card number with 0. For Mastercard cards, you can use any future date for the expiration date.

Test Card Numbers for Successful Network Token Provisioning

Card Brand	Number	Expiration Date	CVV
American Express	Any	Any	Any
Mastercard	512X342233150747	Any	Any
Mastercard	512X343287499758	Any	Any
Mastercard	51203501XXX64594	Any	Any

Test Card Numbers for Successful Network Token Provisioning (continued)

Card Brand	Number	Expiration Date	CVV
Visa	46229431231XX639	12/26	242
Visa	46229431231XX647	12/26	749
Visa	46229431231XX654	12/26	972
Visa	46229431231XX662	12/26	344
Visa	46229431231XX67X	12/26	306
Visa	46229431231XX688	12/26	065
Visa	46229431231XX696	12/26	264

Important: Once a network token has been successfully provisioned for one of the above test PANs there is no way to delete the network token to further attempt successful provisioning. Please be aware of this when testing.

Unsuccessful Network Token Provisioning

Use these test card numbers to test unsuccessful provisioning of network tokens.

For American Express cards, replace the X in the PAN with a 0. For Visa cards, replace the X in the PAN with any number. You can use any future date for the expiration date.

Test Card Numbers for Unsuccessful Network Token Provisioning

Card Brand	PAN	Expiration Date	CVV	Failure Reason
American Express	370000000XXXX28	Any	Any	CARD_NOT_ELIGIBLE
American Express	3700000000XXXX2	Any	Any	DECLINED
American Express	37000000XXXX119	Any	Any	SERVICE_UNAVAILABLE
American Express	370000000XXXX36	Any	Any	CARD_NOT_ALLOWED
Visa	4000000011XXXXXX	Any	Any	CARD_VERIFICATION_FAILED
Visa	4001770011XXXXXX	Any	Any	CARD_NOT_ELIGIBLE

Test Card Numbers for Unsuccessful Network Token Provisioning (continued)

Card Brand	PAN	Expiration Date	CVV	Failure Reason
Visa	4010057011XXXXXX	Any	Any	CARD_NOT_ALLOWED
Visa	4010057022XXXXXX	Any	Any	DECLINED
Visa	4020057022XXXXXX	Any	Any	DECLINED
Visa	4010057033XXXXXX	Any	Any	SERVICE_UNAVAILABLE
Visa	4020057033XXXXXX	Any	Any	SERVICE_UNAVAILABLE
Visa	4010057044XXXXXX	Any	Any	SYSTEM_ERROR
Visa	4020057044XXXXXX	Any	Any	SYSTEM_ERROR
Visa	4020057055XXXXXX	Any	Any	INVALID_REQUEST

Visa Token for Token

Use these Visa test card numbers to test token for token provisioning of network tokens. Replace the X in the card number with any number and use any future date for the expiration date.

Test Card Numbers for Token for Token

Card Brand	Number	Expiration Date	CVV	Response
Mastercard	Any	Any	Any	SUCCESS
Visa	4000010011XXXXXX	Any	Any	CARD_VERIFICATION_FAILED
Visa	4000010022XXXXXX	Any	Any	CARD_NOT_ELIGIBLE
Visa	4000010033XXXXXX	Any	Any	CARD_NOT_ALLOWED
Visa	4000010044XXXXXX	Any	Any	SERVICE_UNAVAILABLE
Visa	4000010055XXXXXX	Any	Any	SYSTEM_ERROR
Visa	4000010088XXXXXX	Any	Any	INVALID_REQUEST

Visa Push Provisioning

Use these Visa account reference ID numbers to test unsuccessful push provisioning of network tokens. To successfully test token provisioning for Visa, you can use any 16-digit alphanumeric account reference ID.

Test Card Numbers for Push Provisioning

Card Brand	Account Reference ID	Response
Visa	Any	Success
Visa	aaaaac907033097c2ec91c3cea9d6d02	cardVerificationFailed
Visa	bbbbbc907033097c2ec91c3cea9d6d02	cardNotEligible
Visa	cccccc907033097c2ec91c3cea9d6d02	cardNotAllowed
Visa	dddddd907033097c2ec91c3cea9d6d02	provisionDataExpired
Visa	ffffff907033097c2ec91c3cea9d6d02	SERVICE_UNAVAILABLE
Visa	gggggg907033097c2ec91c3cea9d6d02	SYSTEM_ERROR

Token Management Service Onboarding

This section contains information necessary to onboard merchants and TMS vault management:

Merchant ID Hierarchy

The Business Center is an online portal provisioned to partners and end merchants. This portal can be used to onboard merchants, view transactional activity and generate and download reports among other things.

There are two environments associated with the Business Center. Each has its own corresponding URL in order to gain access to the Business Center for the relevant environment:

Test: https://businesscentertest.cybersource.com

Production: https://businesscenter.cybersource.com

In order to gain access to Business Center partners/merchants must be provisioned with an Organization ID, otherwise known as a merchant ID (MID). There are multiple types of MIDs:

- **Portfolio**: This is typically a MID that is provisioned to partners. Portfolio MIDs enable partners to onboard merchants into either a test or production environment.
- **Merchant**: This is a parent MID that can house multiple transactional MIDs. This will be directly associated with the end merchant and will be created by the partner under the portfolio MID. This MID will be attached to specific functionality such as the token vault (Token Management Service vault).
- **Transactional**: This is a child MID. Each partner's end merchant may have multiple transactional MIDs. The transactional MID is typically used for processing into Token Management Service, for example, to provision a network token via the Token Management Service API. This will be directly associated with the partners end merchant and will be created by the partner under the portfolio MID.

Merchant ID Registration

A Cybersource MID is a unique value within Cybersource that you define during account registration. Your MID identifies your merchant account and payment configuration within Cybersource systems. You provide this identifier when you sign in to the Business Center and submit transactions to Cybersource.

Multiple MIDs can be configured for various token types. You receive the instrument identifier token regardless of your account's token type. Reasons for multiple MIDs include:

- You have multiple processors.
- Point-of-sale terminals have unique MIDs, which are usually configured for the PAN-only instrument identifier token.

When you have multiple MIDs, you can set up one token vault to which all of your MIDs have access or set up multiple vaults to limit access to tokens. See Token Vault Management (on page 62) for more information on setting up and managing your token vault.

Create an Evaluation Account

To create an evaluation account, visit the Business Center Evaluation Account Sign-Up page.

To complete the registration process, follow the email instructions that you received to activate your merchant account, and log in to the Business Center.

Send your **merchantID** to TMS representative supporting you with integration to create a vault and enable Token Management Service with network tokens.

Portfolio MIDs for Partners

Partners will need to onboard merchants using a portfolio MID. To create a portfolio MID, contact Cybersource support. For information about creating a portfolio MID, visit the Support Center:

http://support.visaacceptance.com

Customer support will respond with a questionnaire. The below information will need to be completed:

- **Organization ID**: Portfolio MID name
- **Environment**: Test and Production
- **Business information**: The business name and address

- **Business contact**: The contact that receives an email registration link to gain access to Business Center through the portfolio MID.
- **Technical contact**: The contact that receives automatically generated notifications, such as product updates, as well as non-urgent notifications.
- **Emergency contact**: The contact that receives urgent messages such as service outage notifications
- **Merchant notifications**: This will send a welcome email to the business contact associated with the end merchant.
- **Processing information**: Not applicable.
- **Product information**: TMS only
- **Customer Support**: Not applicable.
- **Branding**: Not applicable.

Token Vault Management

Token vaults are where merchants store their customer and payment data. A Business Center internal user can enable the TMS vault.

Vaults are assigned to an owner, and all data within the vault belongs to the owner. You can grant permission to individual MIDs to create, retrieve, update, and delete tokens within a vault. Created tokens belong to the owner of the vault, not the creator of the token. If you remove a MID from a vault, it can no longer access any tokens within that vault, including tokens created under that MID.

Important: It is not currently possible to merge vaults, so ensure that merchants are set up with the correct vault by creating a new vault or granting access to an existing vault.

Configure the Token Vault Settings Using the Business Center

Follow these steps to configure your merchant token vault settings:

1. Log in to the Business Center test environment or production environment.
    - **Test:** `https://businesscentertest.cybersource.com`
    - **Production:** `https://businesscenter.cybersource.com`
2. In the left navigation panel, click the **Token Management** icon ().

3. Click **Vault Management New**. The Vault Management page appears.
4. From the Vault Owner drop-down list, select the vault owner..
5. In the Details column, click **Vault Settings**. The Edit Vault page appears.
6. Click **Edit**.
   A dialog box appears with a message to warn you that changing your vault settings could result in your merchants being unable to access tokens, which could result in failing transactions. Click **Yes** if you want to continue.
7. Enter the vault name, supported payment methods, supported token types and formats, card number masking format, payment instrument storing configuration, and the webhook URL. For each token type, you can choose from these token formats:
   - 32 Character Hex
   - 22 Digits
   - 19 Digits Luhn Check Passing
   - 16 Digits Luhn Check Passing

   Important: Account Updater is incompatible with instrument identifier tokens in the 22-digit format.
8. Click **SAVE**.
9. To return to the vault management page, click **VAULT MANAGEMENT**.

Configure the Token Vault Access Using the Business Center

Follow these steps to configure your merchant token vault access settings:

1. Log in to the Business Center test environment or production environment.
   - **Test:** `https://businesscentertest.cybersource.com`
   - **Production:** `https://businesscenter.cybersource.com`
2. In the left navigation panel, click the **Token Management** icon ().
3. Click **Vault Management New**. The Vault Management page appears.
4. Select the vault owner that you want to configure from the Vault Owner drop-down list.
5. In the Details column, click **Access Settings**. The MID Access page appears.

6. Check the box for the vault settings you want to enable for each merchant you want to configure:
    - Visa Token
    - Mastercard Token
    - Card Unmasked
    - Create
    - Update
    - Retrieve
7. Click **Submit** to save your settings.

Configure Network Tokenization Using the Business Center

Follow these steps to configure a merchant token vault network tokenization settings:

1. Log in to the Business Center test environment or production environment.
    - **Test:** `https://businesscentertest.cybersource.com`
    - **Production:** `https://businesscenter.cybersource.com`
2. In the left navigation panel, click the **Token Management** icon ().
3. Click **Vault Management New**. The Vault Management page appears.
4. Select the vault owner that you want to configure from the Vault Owner drop-down list.
5. In the Details column, click **Network Tokenization**. The Network Tokenization page appears.
6. On the VISA tab, switch the **Enroll to VISA Token Services** button to On to enable Visa token services.
   The required business information for the merchant information will be populated:
    - Merchant name
    - Website URL
    - Country code
7. Click **Manage Details**.

a. Check **Enable Visa Token Provisioning** to enable Visa network token provisioning.

b. Check **Enable Visa Token Transactions** to enable Visa transaction processing using network tokens.

8. On the MASTERCARD tab, switch the **Enroll to MASTERCARD Token Services** button to On to enable Visa token services.
9. Click **Manage Details**.

    a. Check **Enable Mastercard Token Provisioning** to enable Mastercard network token provisioning.

    b. Check **Enable Mastercard Token Transactions** to enable Mastercard transaction processing using network tokens.

10. Enter the token requestor ID (TRID) information:

    ◦ TRID (Visa and Mastercard)

    ◦ Relationship Id (Visa only)

11. Click **Submit** to save your settings.

Message-Level Encryption Keys

You must use message-level encryption (MLE) in order for personally identifiable information, such as payment information, to be returned unmasked by TMS. You must create an MLE security key for your Cybersource merchant account in the Business Center before a TMS response can return unmasked payment information using MLE.

MLE keys can be created at the portfolio and transacting levels of an organization. You must create an MLE key at the portfolio level of an organization if you want to use a single MLE key for the encryption and decryption of payment information for multiple merchants. To do so, you must log in to the Business Center using your portfolio credentials and ensure that the MLE key is generated for your organization.

MLE keys expire after 3 years.

Security keys can be used to make any request, including payments. Treat your security keys as you would any secure password.

You must use separate keys for the test and production environments.

Prerequisite

You must have a tool such as OpenSSL installed on your system.

To create an MLE key, you must first extract a public key. You can use a tool such as OpenSSL to extract the key:

```
openssl genrsa -out private.pem 2048 && openssl rsa -in private.pem -outform PEM
 -pubout -out public.pem
```

For information creating an MLE key, see Creating a Message-Level Encryption Key (on page 67).

Creating a Message-Level Encryption Key

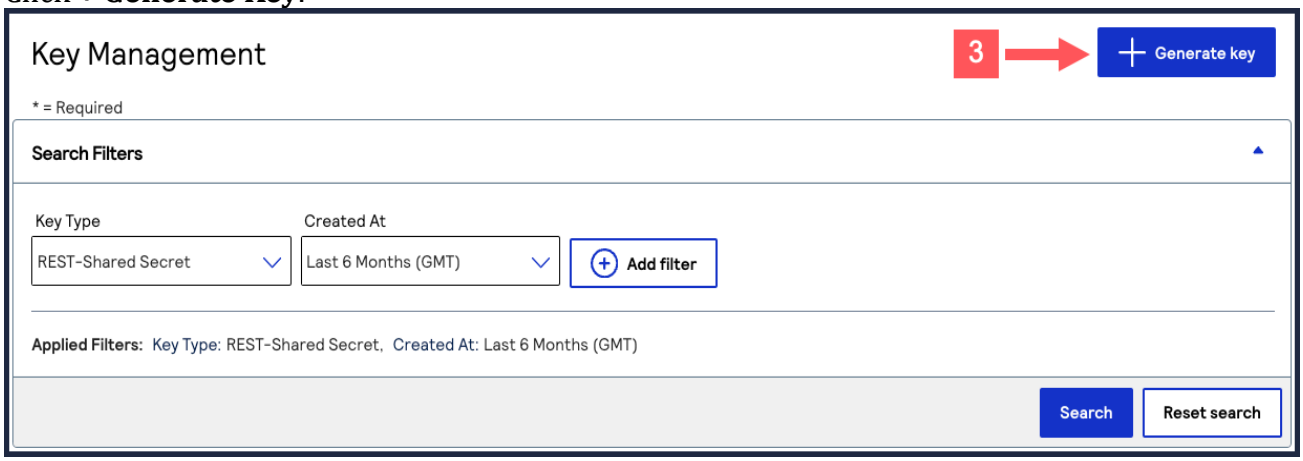
Follow these steps to create a message-level encryption key:

1. Log in to the Business Center:

   `https://businesscentertest.cybersource.com`

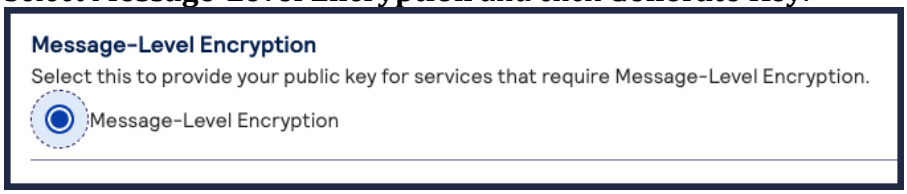
2. On the left navigation panel, choose **Payment Configuration > Key Management**.

3. Click **+ Generate Key**.

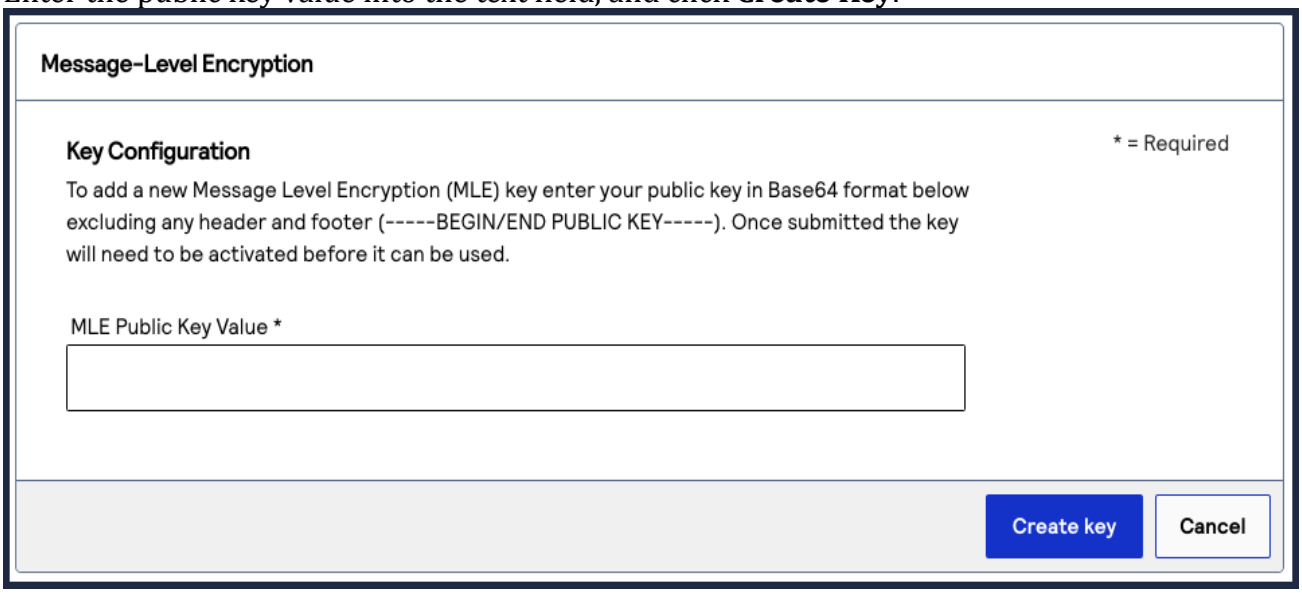


   The Create Key page appears.

4. Select **Message-Level Encryption** and click **Generate Key**.



5. Enter the public key value into the text field, and click **Create Key**.



Network Tokenization

This section contains information on network tokenization:

Network Token Enablement

Network token enablement is currently a manual process and requires a request to be sent to Cybersource support. For more information about network token enablement, visit the Support Center:

http://support.visaacceptance.com

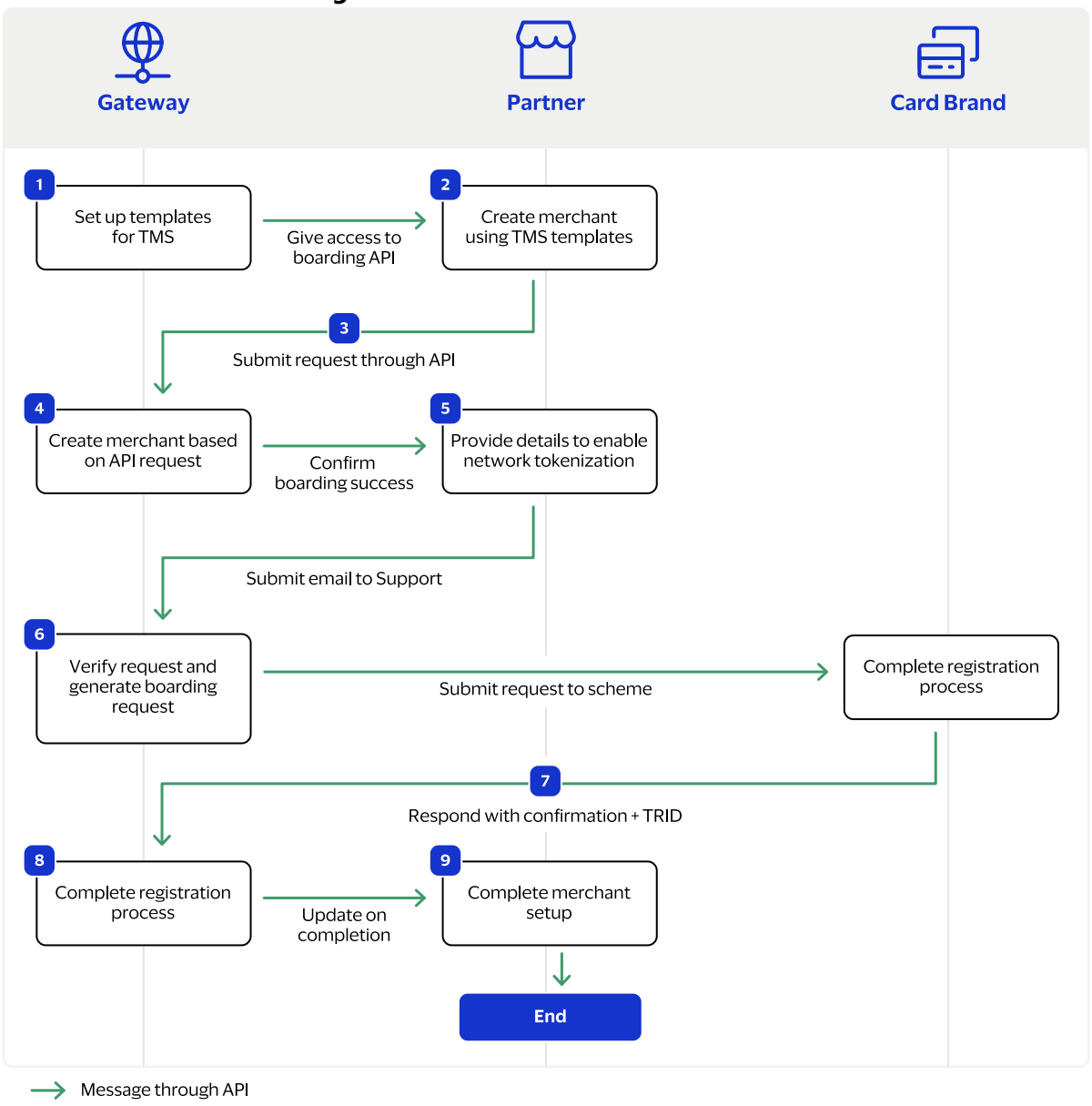
Important: Before sending the request, you must ensure that the merchant/parent MID has been created and the TMS product is enabled.

Network Token Onboarding—Partner Model

This workflow shows the merchant onboarding process.

The workflow begins when the partner creates a merchant profile on the Cybersource platform using TMS templates.

Network Token Onboarding for Partners



1. The gateway sets up templates for TMS.
2. partner creates merchant profile using TMS templates.
3. The partner submits a request to Cybersource via API.
4. Cybersource creates a merchant based on the API request and confirms boarding success with partner.
5. The partner provides merchant details via email to enable network tokenization and submits email to Cybersource support.
6. Cybersource verifies the request, generates the onboarding request, and sends the request to the card brand.

7. The card brand completes the registration process and responds to Cybersource with the confirmation and token requestor ID (TRID).

8. Cybersource notifies the partner that onboarding is complete.

9. The partner completes merchant setup.

Network Token Life-Cycle Management

Life-cycle management is a key feature of credentials-on-file (COF) network tokenization. Issuers can keep COF network tokens updated as changes are made to their cardholders' accounts. TMS notifies you in real time when updates are made to a card represented by the COF network token in your TMS vault. Issuers push the life-cycle management updates either in real time or via a batch process to the card brands. Life-cycle updates and timelines will vary by issuer based on their update process. For example, TMS notifies you when a card becomes inactive.

There are three distinct types of life-cycle management events:

- COF token status changes. Token statuses are:
  - `ACTIVE`: The account and network token are active and in good standing.

    When COF network tokens are active, merchants can process transactions according to their COF agreement.

  - `DELETED`: This is the final state for a network token. A network token can be deleted when the account is closed or on cardholder instruction.

    Merchants must request a new credential from the cardholder.

  - `EXPIRED`: This is a temporary status for COF network tokens where the token has passed its expiration date. This is not a status that results from a life-cycle management update.

    Cybersource does not process transactions with network tokens that have an `EXPIRED` status and merchants should not send transactions with network tokens that have an `EXPIRED` status. When the network token status is `EXPIRED`, there should be a life-cycle management notification to update the status of the token, or the merchant can submit a re-provision.

  - `SUSPENDED`: This status is temporary for COF network tokens and can change to `ACTIVE` or `DELETED`. Merchants should not send authorizations on suspended tokens. However, these tokens can be re-activated by the issuer later. Suspended COF network token events are usually triggered according to cardholder instruction or flagged by the issuer for suspected fraudulent activity. When the status changes from `SUSPENDED` to `ACTIVE` or `DELETED`, a merchant receives a life-cycle management update.

    Merchants can proactively contact a cardholder to update the credential or have them contact the issuer to reactivate the credential.

- PAN updates to the COF network token:

- `NEW PAYMENT ACCOUNT EXPIRY`: A new expiration date has been provided by the issuer associated with the PAN.

  Merchants can retrieve the new expiration date and store it in their cardholder records.

- `NEW PAYMENT ACCOUNT NUMBER`: A new PAN has been provisioned for the network token.

  Merchants can retrieve the new PAN suffix (last four digits) and the new expiration date to store in their cardholder records.

- Life-cycle management reason values:

  - `CARD_UPDATED`: The card expiration date or last four digits have been updated.
  - `METADATA_UPDATE`: The card metadata has been updated.
  - `PROVISIONED`: A network token has been provisioned. This reason value is only available when the **eventType** is set to `tms.networktoken.provisioned`. For more information, see Manage Webhook Subscriptions (on page 78).
  - `TOKEN_STATUS_UPDATE`: The network token status has been updated.
  - `TOKEN_UPDATED`: The token expiration date has been updated.

REST Examples: Life-Cycle Management Notifications

Network Token Enrollment Notification

```
{
 "eventType": "tms.networktoken.provisioned",
 "webhookId": "261d2616-xxxx-9ba8-q3456-8b588d0a5f2a",
 "productId": "tokenManagement",
 "organizationId": "mid",
 "eventDate": "2025-02-24T16:46:27",
 "transactionTraceId": "234234234234234c14e374c2b5a31e26c4316f0dc-0",
 "retryNumber": 0,
 "payload": {
   "data": {
     "reason": "PROVISIONED",
     "id": "2EE7067B2F015632E0631E588E0A1987",
     "type": "tokenizedCardEnrollments",
     "version": "1.0",
     "_links": {
       "tokenized-cards": [
         {
```

```
          "href": "/tms/v2/tokenized-cards/2E8B371F931C4B9BE0631D588D0AF123",
          "id": "2E8B371F931C4B9BE0631D588D0AF123",
          "state": "ACTIVE"
        }
      ],
      "instrumentIdentifiers": [
        {
          "href": "/tms/v1/instrumentidentifiers/7045450003485829123",
          "id": "7045450003485829870"
        }
      ]
    }
  },
  "organizationId": "mid"
 },
 "requestType": "NEW"
}
```

Network Token Update Notification

```
{
 "eventType": "tms.networktoken.updated",
 "webhookId": "261d2616-xxxx-9ba8-q3456-8b588d0a5f2a",
 "productId": "tokenManagement",
 "organizationId": "mid",
 "eventDate": "2025-02-24T16:46:27",
 "transactionTraceId": "234234234234234c14e374c2b5a31e26c4316f0dc-0",
 "retryNumber": 0,
 "payload": {
   "data": {
     "reason": "TOKEN_STATUS_UPDATED",
     "id": "2EE7067B2F015632E0631E588E0A1987",
     "type": "tokenizedCardUpdates",
     "version": "1.0",
     "_links": {
       "tokenized-cards": [
         {
           "href": "/tms/v2/tokenized-cards/2E8B371F931C4B9BE0631D588D0AF123",
           "id": "2E8B371F931C4B9BE0631D588D0AF123",
           "state": "DELETED"
         }
       ],
       "instrumentIdentifiers": [
         {
           "href": "/tms/v1/instrumentidentifiers/7045450003485829123",
           "id": "7045450003485829870"
         }
```

```
        ]
      }
    },
    "organizationId": "mid"
 },
 "requestType": "NEW"
}
```

Network Token Life-Cycle Management Reports

You can generate reports that contain Life-Cycle Management events for Network Tokens. These reports include network token-related fields that are updated as a result of the Life-Cycle Management events sent to the Token Management Service.

For more information about how to generate these reports in the Business Center, see the *Reporting User Guide*.

For more information about how to generate these reports using the Reporting API, see the *Reporting API*.

The *Reporting User Guide* and the *Reporting Developer Guide* contain these relevant topics:

- Downloading Available Reports
- Creating Custom Reports
- Subscribing to Standard Reports
- Fields and Descriptions for Downloadable Reports

Token Requestor IDs

A token requestor ID (TRID) is a unique identifier that entities such as merchants use to request network tokens from token providers. Having a TRID is a prerequisite for enabling network tokenization.

Each entity must register with the token provider to get a TRID. Contact a Cybersource representative to enroll a merchant as a token requestor.

Visa and Mastercard TRIDs

An internal user can enroll a merchant as a VISA or Mastercard token requestor through the Business Center.

Follow these steps to enroll a merchant as a token requestor in the Business Center:

1. Navigate to **Token Management**.
2. Click **Vault Management**.
3. Use the Vault Owner filter to search for the merchant account that has TMS enabled.
4. Choose the merchant account to view the TMS vaults that are configured for the merchant.
5. Click **Network Tokenization**.
6. Click **Enroll to VISA/Mastercard token services**.
7. Enter the required information for each card type:

   Mastercard

   Business entity name

   Visa

   Merchant name

   Merchant website URL

   Merchant country code
8. Click **Onboard with Acquirer ID**.
9. Enter the required information:

   Acquirer ID

   Set the value to `40010052242`. It is a static acquirer ID that is used for TMS.

   Acquirer Merchant ID

   Enter your organization ID.
10. Click **Enroll to Network Token Services** to complete enrollment.

When the enrollment is submitted, the relationship ID and token requestor ID appear on the page for Visa Token Service (VTS) and the token requestor ID appears for Mastercard.

In order to request a TRID from the token provider, Cybersource uses merchant business details already stored. If any of the details are not present, a dialog form should appear prompting you to complete the missing information.

American Express TRIDs

Enrollment as a token requestor for American Express is a manual process. Contact your Cybersource representative to request the TRID for American Express.

Allow 2 to 3 days for the completion of your request.

Important: Service establishment (SE) Numbers are required in order to process American Express card transactions.

Manage Webhook Subscriptions

This section contains information on creating, retrieving, and updating webhook subscriptions. You can create, retrieve, update, or delete notification subscriptions for various events by submitting an HTTP POST, GET, PATCH, or DELETE request to the `notification-subscriptions/v1/webhooks` endpoint. Use the webhooks REST API to:

- Create Keys for Digital Signature (on page 80)
- Create Webhook Subscription (on page 82)
- Retrieve Webhook Subscription Details (on page 85)
- Update Webhook Subscription (on page 87)
- Delete Webhook Subscription (on page 89)

When you send an API request to create a webhooks subscription, you must include the product and its associated events to which you are subscribing.

You can create webhooks subscriptions for these Token Management Service network token events:

Token Management Service

Product ID	Event Types	Description
`tokenManagement`	`tms.networktoken.updated`	Notifies you of a network token's change in expiration date or status (suspend, resume, or deactivate).
	`tms.networktoken.provisioned`	Notifies you when a network token provision for an instrument identifier token has been successful.

Example: Product and Network Token Events in a Webhook Subscription

```
"productId": "tokenManagement",
"eventTypes": [
  "tms.networktoken.provisioned",
  "tms.networktoken.updated"
]
```

For more info, see the Webhooks Developer Guide .

Create Keys for Digital Signature

This section shows you how to create keys for digital signature.

Endpoint

Test: `POST https://apitest.cybersource.com/kms/egress/v2/keys-sym`

Production: `POST https://api.cybersource.com/kms/egress/v2/keys-sym`

Production in India: `POST https://api.in.cybersource.com/kms/egress/v2/keys-sym`

Required Fields for Creating Keys for Digital Signature

clientRequestAction

keyInformation.provider

keyInformation.tenant

keyInformation.keyType

keyInformation.organizationId

REST Example: Creating Keys for Digital Signature

Request

```
{
  "clientRequestAction": "CREATE",
  "keyInformation": {
    "provider": "nrtd",
    "tenant": "testrest",
    "keyType": "sharedSecret",
    "organizationId": "testrest"
  }
}
```

Response to a Successful Request


```
{
    "submitTimeUtc": "2023-02-10T21:26:58Z",
    "status": "SUCCESS",
    "keyInformation": {
        "provider": "nrtd",
        "tenant": "testrest",
        "organizationId": "testrest",
        "keyId": "f4602849-1466-7937-e053-5a588d0ac970",
        "key": "CWK8MHJbHldt74kftIP/+0tTG89We+SWkS7qXjiVJJA=",
        "keyType": "sharedSecret",
        "status": "active",
        "expirationDate": "2026-02-09T21:26:58Z"
    }
}
```

Create Webhook Subscription

This section shows you how to create a webhook subscription.

Important: You must set the **organizationId** field value to that of the TMS vault owner. To receive life-cycle management notifications for network tokens that are created by transacting merchant IDs (MIDs) under that TMS vault, you must set the scope of the field value to `descendants`.

Endpoint

Test: `POST https://apitest.cybersource.com/notification-subscriptions/v1/webhooks`

Production: `POST https://api.cybersource.com/notification-subscriptions/v1/webhooks`

Production in India: `POST https://api.in.cybersource.com/notification-subscriptions/v1/webhooks`

Required Fields for Creating Webhook Subscription

clientRequestAction

keyInformation.provider

keyInformation.tenant

The value must be set to `nrtd`.

keyInformation.keyType

keyInformation.organizationId

keyInformation.expiryDuration

REST Example: Creating a Webhook Subscription

Request

```
{
   "organization": {"organizationId": "TMSVaultOwnerOrgID"},
   "product": {"productId": "tokenManagement"},
   "webhook":    {
```

```
      "webhookId": "e33b4ff7-f94a-2de4-e053-a2588e0a0403",
      "webhookUrl": "https://URL",
      "createdOn": "2021-12-15 23:46:00.053",
      "eventTypes":       [
         {"name": "tms.networktoken.provisioned"},
         {"name": "tms.networktoken.updated"}
      ],
      "status": "ACTIVE",
      "retryPolicy":       {
         "algorithm": "ARITHMETIC",
         "firstRetry": 5,
         "interval": 5,
         "numberOfRetries": 4,
         "deactivateFlag": false,
         "repeatSequenceCount": 4,
         "repeatSequenceWaitTime": 5
      },
      "securityPolicy": [      {
         "digitalSignatureEnabled": "yes",
         "proxyType": "external",
         "security_id": "c05cc30a-ce9b-487f-be38-65ab5977b5bc",
         "security_type": "key"
      }]
   }
}
```

Response to a Successful Request


```
{
  "organizationId": "TMSVaultOwnerOrgID",
  "productId": "terminalManagement",
  "eventTypes": [
    "tms.networktoken.provisioned",
    "tms.networktoken.updated"
  ],
  "webhookId": "e33b4ff7-f94a-2de4-e053-a2588e0a0403",
  "webhookUrl": "https://NewURL",
  "healthCheckUrl": "https://URL",
  "createdOn": "2022-07-07 17:24:05.116",
  "status": "ACTIVE",
  "retryPolicy": {
    "algorithm": "ARITHMETIC",
    "firstRetry": 1,
    "interval": 1,
    "numberOfRetries": 3,
    "deactivateFlag": false,
    "repeatSequenceCount": 0,
    "repeatSequenceWaitTime": 0
  },
  "securityPolicy": {
    "securityType": "KEY",
    "proxyType": "external",
    "digitalSignatureEnabled": "yes"
  },
  "version": "3",
  "deliveryType": "nrtdCentral",
  "notificationScope": "DESCENDANTS"
}
```

Retrieve Webhook Subscription Details

This section shows you how to retrieve webhook subscription details.

Endpoint

Test: `GET https://apitest.cybersource.com/notification-subscriptions/v1/webhooks/{webhookID}`

Production: `GET https://api.cybersource.com/notification-subscriptions/v1/webhooks/{webhookID}`

Production in India: `GET https://api.in.cybersource.com/notification-subscriptions/v1/webhooks/{webhookID}`

Required Field for Retrieving Webhook Subscription Details

webhookID

Include the ID of the webhook you would like to update.

REST Example: Retrieving Webhook Subscription Details

Request

```
GET https://apitest.cybersource.com/notification-subscriptions/v1/webhooks/e33b4ff7-f94a-2de4-e053-a2588e0a0403
```

Response to a Successful Request


```
{
  "organizationId": "testrest",
  "productId": "tokenManagement",
  "eventTypes": [
    "tms.networktoken.provisioned",
    "tms.networktoken.updated"
  ],
  "webhookId": "e33b4ff7-f94a-2de4-e053-a2588e0a0403",
  "webhookUrl": "https://URL",
  "healthCheckUrl": "https://jURL",
  "createdOn": "2022-07-07 17:24:05.116",
  "status": "ACTIVE",
  "retryPolicy": {
    "algorithm": "ARITHMETIC",
    "firstRetry": 1,
    "interval": 1,
    "numberOfRetries": 3,
    "deactivateFlag": false,
    "repeatSequenceCount": 0,
    "repeatSequenceWaitTime": 0
  },
  "securityPolicy": {
    "securityType": "KEY",
    "proxyType": "external",
    "digitalSignatureEnabled": "yes"
  },
  "version": "3",
  "deliveryType": "nrtdCentral",
  "notificationScope": "DESCENDANTS"
}
```

Update Webhook Subscription

This section shows you how to update a webhook subscription.

Endpoint

Test: `PATCH https://apitest.cybersource.com/notification-subscriptions/v1/webhooks/{webhookID}`

Production: `PATCH https://api.cybersource.com/notification-subscriptions/v1/webhooks/{webhookID}`

Production in India: `PATCH https://api.in.cybersource.com/notification-subscriptions/v1/webhooks/{webhookID}`

Required Field for Updating Webhook Subscription

webhookID

Include the ID of the webhook you would like to update.

REST Example: Updating Webhook Subscriptions

Request

```
{
  "description": "Update to my sample webhook",
  "organizationId": "testrest",
  "productId": "terminalManagement",
  "webhookUrl": "https://NewURL"
}
```

Response to a Successful Request


```
{
  "organizationId": "testrest",
  "productId": "terminalManagement",
  "eventTypes": [
    "tms.networktoken.provisioned",
    "tms.networktoken.updated"
  ],
  "webhookId": "e33b4ff7-f94a-2de4-e053-a2588e0a0403",
  "webhookUrl": "https://NewURL",
  "healthCheckUrl": "https://URL",
  "createdOn": "2022-07-07 17:24:05.116",
  "status": "ACTIVE",
  "retryPolicy": {
    "algorithm": "ARITHMETIC",
    "firstRetry": 1,
    "interval": 1,
    "numberOfRetries": 3,
    "deactivateFlag": false,
    "repeatSequenceCount": 0,
    "repeatSequenceWaitTime": 0
  },
  "securityPolicy": {
    "securityType": "KEY",
    "proxyType": "external",
    "digitalSignatureEnabled": "yes"
  },
  "version": "3",
  "deliveryType": "nrtdCentral",
  "notificationScope": "DESCENDANTS"
}
```

Delete Webhook Subscription

This section shows you how to delete a webhook subscription.

Endpoint

Test: `DELETE https://apitest.cybersource.com/notification-subscriptions/v1/webhooks/{webhookID}`

Production: `DELETE https://api.cybersource.com/notification-subscriptions/v1/webhooks/{webhookID}`

Production in India: `DELETE https://api.in.cybersource.com/notification-subscriptions/v1/webhooks/{webhookID}`

Required Field for Deleting a Webhook Subscription

webhookID

Include the ID of the webhook you would like to update.

REST Example: Deleting a Webhook Subscription

Request

```
DELETE https://apitest.cybersource.com/notification-subscriptions/v1/webhooks/{{tms-webhook-id}}
```

Response to a Successful Request

A successful delete response returns an empty `HTTP 204 No Content` status. For more information, see HTTP Status Codes (on page 358).

Customer Tokens

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

Related information

Manage Customer Tokens *(on page 91)*
Payments with Customer Tokens *(on page 109)*

Manage Customer Tokens

This section contains information on managing customer tokens.

The customer token represents customer-related information including details for a payment card or electronic check, billing address, shipping address, and merchant defined data. You can create, retrieve, update, or delete a customer by submitting an HTTP `POST`, `GET`, `PATCH`, or `DELETE` operation to the `tms/v2/customers` endpoint. Use the TMS REST API to:

- Create a customer token (on page 92)
- Retrieve a customer token (on page 95)
- Update a customer's information (on page 97)
- Delete a customer token (on page 99)

For more information on customer tokens, see Customer Tokens (on page 22).

Create a Customer

This section shows you how to create a customer token with no payment details.

Endpoint

Test: `POST https://apitest.cybersource.com/tms/v2/customers`

Production: `POST https://api.cybersource.com/tms/v2/customers`

Production in India: `POST https://api.in.cybersource.com/tms/v2/customers`

Required Fields for Creating a Customer

You can include any of the following fields in the body of the request:

buyerInformation.merchantCustomerID

buyerInformation.email

clientReferenceInformation.code

merchantDefinedInformation.name

merchantDefinedInformation.value

Related Information

- API Field Reference for the REST API

REST Example: Creating a Customer

Request

```
{
  "buyerInformation": {
    "merchantCustomerID": "Your customer identifier",
    "email": "test@cybs.com"
  },
  "clientReferenceInformation": {
    "code": "123456"
```

```
  },
  "merchantDefinedInformation": [
    {
      "name": "data1",
      "value": "Your customer data"
    }
  ]
}
```

Response to a Successful Request


```
{
  "_links": {
    "self": {
      "href": "/tms/v2/customers/F2F3ADA770102B51E053A2598D0A9078"
    },
    "paymentInstruments": {
      "href":
 "/tms/v2/customers/F2F3ADA770102B51E053A2598D0A9078/payment-instruments"
    },
    "shippingAddresses": {
      "href":
 "/tms/v2/customers/F2F3ADA770102B51E053A2598D0A9078/shipping-addresses"
    }
  },
  "id": "F2F3ADA770102B51E053A2598D0A9078",
  "buyerInformation": {
    "merchantCustomerID": "Your customer identifier",
    "email": "test@cybs.com"
  },
  "clientReferenceInformation": {
    "code": "TC50171_3"
  },
  "merchantDefinedInformation": [
    {
      "name": "data1",
      "value": "Your customer data"
    }
  ],
  "metadata": {
    "creator": "testrest"
  }
}
```

Retrieve a Customer

This section shows you how to retrieve a customer token.

Endpoint

Test: `GET https://apitest.cybersource.com/tms/v2/customers/{customerTokenId}`

Production: `GET https://api.cybersource.com/tms/v2/customers/{customerTokenId}`

Production in India: `GET https://api.in.cybersource.com/tms/v2/customers/{customerTokenId}`

The *{customerTokenId}* is the customer token ID returned in the **id** field when you created the customer token. For more information, see Create a Customer (on page 92).

REST Example: Retrieving a Customer

Request

```
GET https://apitest.cybersource.com/tms/v2/customers/F2F3ADA770102B51E053A2598D0A9
078
```

Response to a Successful Request


```
{
  "_links": {
    "self": {
      "href": "/tms/v2/customers/F2F3ADA770102B51E053A2598D0A9078"
    },
    "paymentInstruments": {
      "href":
 "/tms/v2/customers/F2F3ADA770102B51E053A2598D0A9078/payment-instruments"
    },
    "shippingAddresses": {
      "href":
 "/tms/v2/customers/F2F3ADA770102B51E053A2598D0A9078/shipping-addresses"
    }
  },
  "id": "F2F3ADA770102B51E053A2598D0A9078",
  "buyerInformation": {
    "merchantCustomerID": "Your customer identifier",
    "email": "test@cybs.com"
  },
  "clientReferenceInformation": {
    "code": "TC50171_3"
  },
  "merchantDefinedInformation": [
    {
      "name": "data1",
      "value": "Your customer data"
    }
  ],
  "metadata": {
    "creator": "testrest"
  }
}
```

Update a Customer

This section shows you how to update a customer token.

Endpoint

Test: `PATCH https://apitest.cybersource.com/tms/v2/customers/{customerTokenId}`

Production: `PATCH https://api.cybersource.com/tms/v2/customers/{customerTokenId}`

Production in India: `PATCH https://api.in.cybersource.com/tms/v2/customers/{customerTokenId}`

The *{customerTokenId}* is the customer token ID returned in the **id** field when you created the customer token. For more information, see Create a Customer (on page 92). Include only the fields you want to add or update in the request.

Optional Fields for Updating a Customer

You can include any of the following fields in the body of the request:

buyerInformation.merchantCustomerID

buyerInformation.email

clientReferenceInformation.code

merchantDefinedInformation.name

merchantDefinedInformation.value

Related Information

- API Field Reference for the REST API

REST Example: Updating a Customer

Request

```
PATCH https://apitest.cybersource.com/tms/v2/customers/F2F3ADA770102B51E053A2598D0
A9078
```

Response to a Successful Request


```
{
  "_links": {
    "self": {
      "href": "/tms/v2/customers/F2F3ADA770102B51E053A2598D0A9078"
    },
    "paymentInstruments": {
      "href":
 "/tms/v2/customers/F2F3ADA770102B51E053A2598D0A9078/payment-instruments"
    },
    "shippingAddresses": {
      "href":
 "/tms/v2/customers/F2F3ADA770102B51E053A2598D0A9078/shipping-addresses"
    }
  },
  "id": "F2F3ADA770102B51E053A2598D0A9078",
  "buyerInformation": {
    "merchantCustomerID": "Your customer identifier",
    "email": "test@cybs.com"
  },
  "clientReferenceInformation": {
    "code": "TC50171_3"
  },
  "merchantDefinedInformation": [
    {
      "name": "data1",
      "value": "Your customer data"
    }
  ],
  "metadata": {
    "creator": "testrest"
  }
}
```

Delete a Customer

This section shows you how to delete a customer token.

Endpoint

Test: `DELETE https://apitest.cybersource.com/tms/v2/customers/{customerTokenId}`

Production: `DELETE https://api.cybersource.com/tms/v2/customers/{customerTokenId}`

Production in India: `DELETE https://api.in.cybersource.com/tms/v2/customers/{customerTokenId}`

The *{customerTokenId}* is the customer token ID returned in the **id** field when you created the customer token. For more information, see Create a Customer (on page 92).

Required Fields for Deleting a Customer Token

customerTokenId

Include the ID of the customer token you want to retrieve in the URL path.

Related Information

- API Field Reference for the REST API

REST Example: Deleting a Customer

Request

```
DELETE https://apitest.cybersource.com/tms/v2/customers/F2F3ADA770102B51E053A2598D
0A9078
```

Response to a Successful Request

A successful delete response returns an empty `HTTP 204 No Content` status. For more information, see HTTP Status Codes (on page 358).

Retrieve a Customer's Default Payment with an Unmasked Card Number

This section shows you how to retrieve a customer's default payment with an unmasked card number.

Important: To retrieve unmasked payment details, you must ensure that your MLE key pair and your token vault are configured correctly. For more information on MLE keys, see Message-Level Encryption Keys (on page 65). For more information on token vaults, see Token Vault Management (on page 62). If necessary, contact your Cybersource account manager or customer support.

The response is BASE 64-encoded JSON web encryption (JWE) token. The decoded JWE has these elements:

```
{ "alg": "RSA-OAEP-256", //The algorithm used to encrypt the CEK
    "cty": "json", //The content type
    "typ": "JWT", //The token type
    "enc": "A256GCM", //The algorithm that is used to encrypt the message
    "kid": "keyId" //The serial number of shared public cert for encryption of CEK
}
<Encrypted Data> //The encrypted payload that matches the JSON response normally
 returned by the TMS API, except with an unmasked payment details
```

Header Configuration

You must pass this request header to retrieve unmasked payment details: `Accept: application/jose`.

The term `application/jose` refers to Javascript Object Signing and Encryption (JOSE). JOSE is a framework that provides end-to-end security to JavaScript Object Notation (JSON)-based data structures. JOSE achieves this by offering a collection of specifications to encrypt and digitally sign JSON payloads. In this case, the response is message-level encrypted using a JSON Web Token (JWT).

Endpoint

Test: `GET https://apitest.cybersource.com/tms/v2/customers/{customerTokenId}`

Production: `GET https://api.cybersource.com/tms/v2/customers/{customerTokenId}`

Production in India: `GET https://api.in.cybersource.com/tms/v2/customers/{customerTokenId}`

The *{customerTokenId}* is the customer token ID returned in the **id** field when you created the customer token. For more information, see Create a Customer (on page 92).

REST Example: Retrieving a Customer's Default Payment with an Unmasked Card Number

Request

```
GET https://apitest.cybersource.com/tms/v2/customers/F45FB3E443AC3C57E053A2598D0A9
CFF
```

Response to a Successful Request


```
eyJraWQiOiJiYTE1ZDRmMTIzMTM0NjlkZjg5MDM1Nzk2YWE4Nzc4ZGM0NTY4ODlkIiwiY3R5IjoianNvbi
IsInR5cCI6IkpXVCIsImVuYyI
6IkEyNTZHQ00iLCJhbGciOiJSU0EtT0FFUC0yNTYifQ.zxPTNWvHt40Dbtwlx2T53Jd-vazzEeN7v_6nyy
PE8FpAylO9dMCQL0XOG_1AQZR
ZPhrAvilhV2Gp8xc1OuF6w0w8LtDQGcgoTVQM0HitMXSs05b_0FzYNZXHr9OPwmJxzizNoptI-Arlw55yf
JNM8QNBLYGIEJkKI061P84Dk9
by-c3bfo8z8D0xO4LpA51dndSkk5QFIWaNSz5CC0nuANyPPJzGVLguBx8HYNLMH4g_dx6SEVw-QYBO0-s_
Xfmv3wRjGpH0STzn7j_1MxpW7
tfXaYcrglPNCoOiSHc6dg141lHGSdvALS71qZTo49WVWbuO2kBYyhxVD1x_1P4ztQ.D1H34AH9Rwu1cr5F
.dqCwRR8Ug8uv1Ow437anK8br
Ye7KPWfcw_R7ShoIlNhWCmQcQ9mVK8UCKcbuVbxt7S6_whJHOfWlm1jqwIvA7ZYtHfVyVHsG11wRZZd3vn
4HGJ4rAUa3T0d_EnvF2Jeffpj
cTG6MZN5_nB4z2Ism4dLcxsnWIdzU2993hscS5641wvX3GYAhqD5OqweT1hqW8URyuSQh27WDJSlMmARE0
s3hVq6O0XcdejOmulyVKMNFFp
Wpcif3G-VnTMzDI7iMx448u2tClA895cvG-E4ISDvRZ3eIjH4wgRE5Btxy5SwbS7VfCYDyRLa9z1LewRV3
EwFxvb6_POtq2Da5QYSG7U-XO
aSNie6bZ7oTYKn7lD-1-crcfQY6ieSWUxMKcsi3bD0_yz1Y_Lc0Wc4M7MCIRwDmbctmZvxZuRwiBiSMsKl
l9gQdKTn8sEeGv7DWooJBxqiZ
kbPlzkHnw2El4Z9HETIGH1Iq0nsKC78NzTiT96i0SHN22iqZGYdIUPwB9zZJQGJNxZ3ag_Cf4C6ATAubJG
4jVtdQ_JtbhHLYwhXlQFTiPMQ
rqnoh7GZDaOX3jEP9_LQiNam8U-ZNGuQby2jgqPyLQKb4dsB31eBz9TLCa7SkXqWp5_a18QVpNxeQEB0mJ
C7s0iy3XXB7GxxzrKLgqsxhmJ
ZjLaKUo1-Sen3HG_9oWTlXXh2r2C0b3AzX1HQ8POC5E8RcK7e-tLtvMJrLNHMZXRVAeREbVah13b1Fs-CI
DWnw9QcCUxjfYNCwwOZUAZTIX
tq4YnaQYkSE6OJV-yRrajbo7CzM1HXZEioW9S9eFRJmqKpu_vtatwJXme3XjyjAqahSYCBtIx2__8688Mp
TmSm1_WEOZrNXoV0-htOqQAQ4
IIza4FpWfpevUJUs8hOu2FIr_Adm0-IIE2MsSptXrNvxmahuDwNpmapNUeLg7aZoQVlp33TrwcQ0AfQeD3
6s1nkWOhRmPfjVfXvKCeJxmD2
ndXBJxLzko0BkqWwe9WSH-RDyaN3u7TXHgyp2EDO0p6rRO8F2veRv76T6ppMrGWK9xjZLFaA1kNNt_RTls
n1hedb_R-ztOw-4y4u405yv8E
2z0W0mE58FLlWJ7AORSQjEtVANrlnwzLjE4xi_xv41CrwI8_uhlyQFfy0aj13bOyFg5nQd7k1MsawzFJfs
NBRrCNwf07cpP_4nH_yv4UvV4
qSaf1DR0epfQ4iWvaZOWqhN_vJeAxXAasChiuZJ8gN2qrq_p1fO4lGIUjLcyvn9W1fqDVcgLLZUVCcHdHi
FpZT7C6iKqDQBjUaAQ6E-jsES
0t2EmwkkJ21jxnwUtwCQLShXNUjXDjwKrbAiFAiHrBALXJnBY5Yd0JkJ7srB4dWR4lZxb5M5g9n-gUkto8
O_u1JtII34EgbKbOHbpbOgCEN
rGUnjaIXrTVHy1uWBBTV7gmkkqd1HD9lwqbmj4IDyIdUwjkUJ4ZX9MB_E7PJ_asWnye5-RO3vFlL7_dSQk
mJAiseNrve0bTlynMZW-KsC9s
0We-jjDqJP52uZ-jmJwL1hhIUef3Yt40IytfuyxiIWkGJB0qM2cppdh6F8bdbFxomWeOxFW8qllZ-1hhp5
ISoW6oy8f6OyvIxFQMjkXMfoh
CdBwB3wOZYoCPq2JmrXYExQcStI3MHMkvBkNP5slTX-uT-mJlx5XHLLOPdO9io58D0CEcmdDAiZqPtQssa
FKsj4Z6EMlgewmC8E7mO6LLhF
Fcof_qDaJ7d6ZNnUiVUq0tAQnxiBH6NfKybQ64O_j2yq4bE1Fzc_aOM5YcQqch8riYU0VWeS0xzgdetta8
AYTjmIGE1SKhvEh_KxngNK6Rn
4GiE_-g6tQaI65ZJ9OMfVdzVbuBJ04IU1VI9KPZhAAd_xZYOnKzkaKHlogkMOaI-Pz-CKP4Ij-hrjzNIHH
I8dHtXcLT5e4BGFksN87UGvJu
```

```
PDJEqpCOXcvxRKe5C0129SSMGq_MefP04pkHAnKQ1qg_gIlxz7H2lvCKGBrgnm6yfE1kD9CYqtHdDZlgcZ
Y_dl56MRpW8fOs5xtaDVgTlme
kd4qtt59R6FpN1LQFmpKIqza5AJqPinUaZJSxvF6nkK76xx8ozxFIFitygAkK6eh8fTiuArXkTul0E3277
fg-gCv9h2xz0CnDnNV5ubLfJp
2QDboy-JRE_NFN3E0eqr5MkGETtiXeGyQTGwFtr0KuvsZu4V8qg1DxiF_pPdlszTyhGL2q21Vcr2IBDzrg
NKkGDLPvPXUVIjHA1XM-4dnv9
ZIx0Eb5jsGQhecQrmQaH4wcM3ZE2sWPGwLJutbVuywSnQg6YWz-PAQDk2Er4icNuybTrdw4RoZPNzY-2BG
xWrpbo4J4SMK84jmsxPatCI1s
I12uhXp27LrtHEUfHHLVlFw2KHIpCWirQ7mLicp7Be1dAx3ak-RapHaH9qTVkDuMVuWIzvaj6ulY2mBltc
SMyFpr7_vEGYY3LEU8_Udnvyn
SwKuXqt3MiHE0h2bEeL6X6YXC2D8iOEgsIEh6naEyKxhdfzk0BpM1MyLnAqkr34BoJhyrM_P0ZfRF4YNau
qVqvr0qAZN757nRwHcPfDal22
jnVdxd65TRUtymqV5gnfKNVBFF5NfjU0zcuK3kr3llkSHaULobBOlJ3W-tBx37-0cjuXqci_ZgbRnVCbF3
TnOe_kULrPLrGakcjOfVOXNhg
ckSR6Rz3At7MIhzAhKCtZm3qyxcARxrMGMrIcW3ShE62DfOOlTLotq9TJCkyI6LI93TBtQOlYnDZNcU4KW
9hFqo0ZACy9cbEHA_LNUFtwUL
spFw_gu3AlUTFp3LDRDAGu9_4Ip-aw4xCPOWN7-oNuzfpasdFx7IioHahxvBi1HmMUsn09p96finQndMoB
yC8enwwBILNKJ9BBEgS0jaQn7
Ymag4G1xArMNnECF2ip7CDWc9dvivhHxS_AftERedVYksH7XP3YEMUoOlsOrwVhGGArZrWpHmNcX0woTvK
9lJLjxHMl9w6lhhcvY3X-glV5
AKJczoDnVJe-6Q9cnH0BYE8b_0LgdlcN7dVGEXkjR9EAm0bsytEO6zm53u-zRwq5wzBVBA7WMBuQLPjyvU
B9WPueBfKPhJhnYIYKCZZplRw
RxGg04RKkwl8nKGMSfpITD0L6NiYWg-aS7aSQVa21RpYxZoCr9t2lFo8gxe0Lhr5mGZRLv_toZ5wuxViHU
PTvtG-UVv6IS3M4k6GTzSh90j
OOBeDfChJzPLXzQWLsIYTUfzmEbkcncuo7c8auEgEabcfo4q7loiCuK_QODY_wB6_PDh6rWhINsAN0KC27
sPcv0rIkADo9uDGhKCPb314EK
9RhUUsGBJjOvxX1oKfy0OXfURdYeG8DC6zCZtazrX6DO12rcsZlCPu2Fj1ZPWoAdKYqUAaeX8DdYBAFhvm
hxuLlmXYW4zPj89ZhDrbSDCxs
e0w6rlWbXTaEkiqc3-4S8Az3DJG73jSMB58PtAKUHfpjWR9sp0TLtpfxw_XPtwbE_7EHmchQqNq9zFiB0F
6Cxu1eF-5eObABh-EEpQ68Ppp
3zuorFSSNUuW-nKGl_Eio6gPyUYuMSen8zA.BARlPgBMj068Dt6OGEiPnA
```

Retrieve a Customer's Default Payment and Shipping Details

This section shows you how to retrieve a customer's default payment and shipping details.

Endpoint

Test: `GET https://apitest.cybersource.com/tms/v2/customers/{customerTokenId}`

Production: `GET https://api.cybersource.com/tms/v2/customers/{customerTokenId}`

Production in India: `GET https://api.in.cybersource.com/tms/v2/customers/{customerTokenId}`

The *{customerTokenId}* is the customer token ID returned in the **id** field when you created the customer token. For more information, see Create a Customer (on page 92).

REST Example: Retrieving a Customer's Default Payment and Shipping Details

Request

```
GET https://apitest.cybersource.com/tms/v2/customers/F45FB3E443AC3C57E053A2598D0A9
CFF
```

Response to a Successful Request


```
{
    "_links": {
        "self": {
            "href": "/tms/v2/customers/F45FB3E443AC3C57E053A2598D0A9CFF"
        },
        "paymentInstruments": {
            "href":
 "/tms/v2/customers/F45FB3E443AC3C57E053A2598D0A9CFF/payment-instruments"
        },
        "shippingAddresses": {
            "href":
 "/tms/v2/customers/F45FB3E443AC3C57E053A2598D0A9CFF/shipping-addresses"
        }
    },
    "id": "F45FB3E443AC3C57E053A2598D0A9CFF",
    "clientReferenceInformation": {
        "code": "TC50171_3"
    },
    "defaultPaymentInstrument": {
        "id": "F45FC6785E3C31A2E053A2598D0A5346"
    },
    "defaultShippingAddress": {
        "id": "F45FB3E443AF3C57E053A2598D0A9CFF"
    },
    "metadata": {
        "creator": "testrest"
    },
    "_embedded": {
        "defaultPaymentInstrument": {
            "_links": {
                "self": {
                    "href":
 "/tms/v2/customers/F45FB3E443AC3C57E053A2598D0A9CFF/payment-instruments/F45FC6785
E3C31A2E053A2598D0A5346"
                },
                "customer": {
                    "href": "/tms/v2/customers/F45FB3E443AC3C57E053A2598D0A9CFF"
                }
            },
            "id": "F45FC6785E3C31A2E053A2598D0A5346",
            "default": true,
            "state": "ACTIVE",
            "card": {
                "expirationMonth": "12",
                "expirationYear": "2031",
                "type": "001"
            },
```

```
            "buyerInformation": {
                "currency": "USD"
            },
            "billTo": {
                "firstName": "JOHN",
                "lastName": "DEO",
                "address1": "201 S. Division St.",
                "address2": "Address 2",
                "locality": "Ann Arbor",
                "administrativeArea": "MI",
                "postalCode": "48104-2201",
                "country": "US",
                "email": "test@cybs.com",
                "phoneNumber": "999999999"
            },
            "processingInformation": {
                "billPaymentProgramEnabled": false
            },
            "instrumentIdentifier": {
                "id": "7030000000014911515"
            },
            "metadata": {
                "creator": "testrest"
            },
            "_embedded": {
                "instrumentIdentifier": {
                    "_links": {
                        "self": {
                            "href":
"/tms/v1/instrumentidentifiers/7030000000014911515"
                        },
                        "paymentInstruments": {
                            "href":
"/tms/v1/instrumentidentifiers/7030000000014911515/paymentinstruments"
                        }
                    },
                    "id": "7030000000014911515",
                    "object": "instrumentIdentifier",
                    "state": "ACTIVE",
                    "tokenizedCard": {
                        "state": "ACTIVE",
                        "number": "489537XXXXXX5914",
                        "expirationMonth": "12",
                        "expirationYear": "2022",
                        "type": "visa",
                        "requestorId": "40010052236",
                        "card": {
                            "suffix": "1515",
                            "expirationMonth": "12",
```

```
                            "expirationYear": "2031"
                        }
                    },
                    "card": {
                        "number": "489537XXXXXX1515"
                    },
                    "issuer": {
                        "paymentAccountReference": "V0010013019326121174070050420"
                    },
                    "processingInformation": {
                        "authorizationOptions": {
                            "initiator": {
                                "merchantInitiatedTransaction": {
                                    "previousTransactionId": "123456789619999"
                                }
                            }
                        }
                    },
                    "metadata": {
                        "creator": "testrest"
                    }
                }
            }
        },
        "defaultShippingAddress": {
            "_links": {
                "self": {
                    "href":
 "/tms/v2/customers/F45FB3E443AC3C57E053A2598D0A9CFF/shipping-addresses/F45FB3E443
AF3C57E053A2598D0A9CFF"
                },
                "customer": {
                    "href": "/tms/v2/customers/F45FB3E443AC3C57E053A2598D0A9CFF"
                }
            },
            "id": "F45FB3E443AF3C57E053A2598D0A9CFF",
            "default": true,
            "shipTo": {
                "firstName": "JOHN",
                "lastName": "DEO",
                "company": "Visa",
                "address1": "201 S. Division St.",
                "address2": "Address 2",
                "locality": "Ann Arbor",
                "administrativeArea": "MI",
                "postalCode": "48104-2201",
                "country": "US"
            },
            "metadata": {
```

```
                "creator": "testrest"
            }
        }
    }
}
```

Payments with Customer Tokens

This section contains information on making payments with customer tokens.

The customer token represents customer-related information including details for a payment card or electronic check, billing address, shipping address, and merchant defined data.

You can make a payment using an existing customer token or create one. To make a payment using a new customer token, you must include token creation in the authorization request. For example:

- Create a Customer Token with Validated Payment Details (on page 110)

To process a payment using an existing customer token, you must include the customer token ID as the value in the `paymentInformation.customer.id` field. For example:

- Authorizing a Payment with a Customer Token (on page 115)
- Making a Credit with a Customer Token (on page 122)

For more information on customer tokens, see Customer Tokens (on page 22).

Create a Customer Token with Validated Payment Details

This section shows you how to create a customer with validated payment details.

Endpoint

Test: `POST https://apitest.cybersource.com/pts/v2/payments`

Production: `POST https://api.cybersource.com/pts/v2/payments`

Production in India: `POST https://api.in.cybersource.com/pts/v2/payments`

Required Fields for Creating a Customer Token with Validated Payment Details Using the REST API

orderInformation.amountDetails.currency

orderInformation.amountDetails.totalAmount

orderInformation.billTo.address1

orderInformation.billTo.administrativeArea

orderInformation.billTo.country

orderInformation.billTo.email

orderInformation.billTo.firstName

orderInformation.billTo.lastName

orderInformation.billTo.locality

orderInformation.billTo.postalCode

paymentInformation.card.expirationMonth

paymentInformation.card.expirationYear

paymentInformation.card.number

paymentInformation.card.type

processingInformation.actionList

Set the value to `TOKEN_CREATE`.

processingInformation.actionTokenTypes

Set the value to `customer`.

Related Information

- API Field Reference for the REST API

REST Example: Creating a Customer Token with Validated Payment Details

Request

```
POST https://apitest.cybersource.com/pts/v2/payments

{
    "clientReferenceInformation": {
        "code": "TC50171_3"
    },
    "processingInformation": {
        "commerceIndicator": "internet",
        "actionList": [
            "TOKEN_CREATE"
        ],
        "actionTokenTypes": [
            "customer",
            "paymentInstrument",
            "shippingAddress"
        ]
    },
    "orderInformation": {
        "billTo": {
            "country": "US",
            "lastName": "Deo",
            "address2": "Address 2",
            "address1": "201 S. Division St.",
            "postalCode": "48104-2201",
            "locality": "Ann Arbor",
            "administrativeArea": "MI",
            "firstName": "John",
            "phoneNumber": "999999999",
            "district": "MI",
            "buildingNumber": "123",
            "company": "Visa",
            "email": "test@cybs.com"
        },
        "shipTo": {
            "country": "US",
            "lastName": "Deo",
```

```
            "address2": "Address 2",
            "address1": "201 S. Division St.",
            "postalCode": "48104-2201",
            "locality": "Ann Arbor",
            "administrativeArea": "MI",
            "firstName": "John",
            "phoneNumber": "999999999",
            "district": "MI",
            "buildingNumber": "123",
            "company": "Visa",
            "email": "test@cybs.com"
        },
        "amountDetails": {
            "totalAmount": "102.00",
            "currency": "USD"
        }
    },
    "paymentInformation": {
        "card": {
            "expirationYear": "2031",
            "number": "4895379987X11515",
            "securityCode": "089",
            "expirationMonth": "12",
            "type": "001"
        }
    }
}
```

Response to a Successful Request


```
{
    "_links": {
        "authReversal": {
            "method": "POST",
            "href": "/pts/v2/payments/676063008813612730395 5/reversals"
        },
        "self": {
            "method": "GET",
            "href": "/pts/v2/payments/67606300881361273039 55"
        },
        "capture": {
            "method": "POST",
            "href": "/pts/v2/payments/676063008813612730395 5/captures"
        }
    },
    "clientReferenceInformation": {
        "code": "TC50171_3"
    },
    "id": "6760630088136127303955",
    "orderInformation": {
        "amountDetails": {
            "authorizedAmount": "102.00",
            "currency": "USD"
        }
    },
    "paymentAccountInformation": {
        "card": {
            "type": "001"
        }
    },
    "paymentInformation": {
        "tokenizedCard": {
            "type": "001"
        },
        "card": {
            "type": "001"
        }
    },
    "pointOfSaleInformation": {
        "terminalId": "111111"
    },
    "processorInformation": {
        "paymentAccountReferenceNumber": "V0010013019326121174070050420",
        "approvalCode": "888888",
        "networkTransactionId": "123456789619999",
        "transactionId": "123456789619999",
        "responseCode": "100",
```

```
        "avs": {
            "code": "X",
            "codeRaw": "I1"
        }
    },
    "reconciliationId": "69816012FDTK35GM",
    "status": "AUTHORIZED",
    "submitTimeUtc": "2023-02-10T21:03:29Z",
    "tokenInformation": {
        "instrumentidentifierNew": false,
        "instrumentIdentifier": {
            "state": "ACTIVE",
            "id": "7030000000014911515"
        },
        "shippingAddress": {
            "id": "F45FB3E443AF3C57E053A2598D0A9CFF"
        },
        "paymentInstrument": {
            "id": "F45FC6785E3C31A2E053A2598D0A5346"
        },
        "customer": {
            "id": "F45FB3E443AC3C57E053A2598D0A9CFF"
        }
    }
}
```

Authorizing a Payment with a Customer Token

This section provides the information you need to authorize a payment with a customer token.

Endpoint

Production: `POST https://api.cybersource.com/pts/v2/payments`

Test: `POST https://apitest.cybersource.com/pts/v2/payments`

Production in India: `POST https://api.in.cybersource.com/pts/v2/payments`

Required Fields for Authorizing a Payment with a Customer Token

clientReferenceInformation.code

orderInformation.amountDetails.currency

orderInformation.amountDetails.totalAmount

paymentInformation.customer.id

Set to the ID of the customer token you want to use.

Related Information

- API field reference guide for the REST API

REST Example: Authorizing a Payment with a Customer Token

Request

```
{
  "clientReferenceInformation": {
    "code": "12345678"
  },
    "paymentInformation": {
        "customer": {
            "id": "F45FB3E443AC3C57E053A2598D0A9CFF"
        }

    },
```

```
    "orderInformation": {
        "amountDetails": {
            "currency": "USD",
            "totalAmount": "10.00"
        }
    }
}
```

Response to a Successful Request

The request response returns the payment instrument and shipping address IDs that are used as the customer's defaults.


```
{
  "_links": {
    "authReversal": {
      "method": "POST",
      "href": "/pts/v2/payments/705592887155681810 4953/reversals"
    },
    "self": {
      "method": "GET",
      "href": "/pts/v2/payments/705592887155681810 4953"
    },
    "capture": {
      "method": "POST",
      "href": "/pts/v2/payments/705592887155681810 4953/captures"
    }
  },
  "clientReferenceInformation": {
    "code": "12345678"
  },
  "id": "705592887155681810 4953",
  "orderInformation": {
    "amountDetails": {
      "authorizedAmount": "10.00",
      "currency": "USD"
    }
  },
  "paymentAccountInformation": {
    "card": {
      "type": "001"
    }
  },
  "paymentInformation": {
    "tokenizedCard": {
      "type": "001"
    },
    "instrumentIdentifier": {
      "id": "7010000000016241111",
      "state": "ACTIVE"
    },
    "shippingAddress": {
      "id": "0F35F0D99AD088B5E063A2598D0AE066"
    },
    "paymentInstrument": {
      "id": "0F35E9CFEA463E34E063A2598D0A3FC2"
    },
```

```
    "card": {
      "type": "001"
    },
    "customer": {
      "id": "B21E6717A6F03479E05341588E0A303F"
    }
  },
  "pointOfSaleInformation": {
    "terminalId": "111111"
  },
  "processorInformation": {
    "approvalCode": "888888",
    "networkTransactionId": "123456789619999",
    "transactionId": "123456789619999",
    "responseCode": "100",
    "avs": {
      "code": "X",
      "codeRaw": "I1"
    }
  },
  "reconciliationId": "67467352CRIISD1G",
  "status": "AUTHORIZED",
  "submitTimeUtc": "2024-01-18T15:48:07Z"
}
```

REST Example: Authorizing a Payment Using a Customer Token Linked to a Network Token

Request

```
{
  "clientReferenceInformation": {
    "code": "12345678"
  },
  "paymentInformation": {
    "customer": {
      "id": "F60328413BAB09A4E053AF598E0A33DB"
    }
  },
  "orderInformation": {
    "amountDetails": {
      "totalAmount": "102.21",
      "currency": "USD"
```

```
    }
  }
}
```

Response to a Successful Request

The request response returns the payment instrument and shipping address IDs that are used as the customer's defaults.


```
{
  "_links": {
    "authReversal": {
      "method": "POST",
      "href": "/pts/v2/payments/677864707112638490495 3/reversals"
    },
    "self": {
      "method": "GET",
      "href": "/pts/v2/payments/6778647071126384904953"
    },
    "capture": {
      "method": "POST",
      "href": "/pts/v2/payments/6778647071126384904953/captures"
    }
  },
  "clientReferenceInformation": {
    "code": "TC50171_3"
  },
  "id": "6778647071126384904953",
  "issuerInformation": {
    "responseRaw": "0110322000000E100002000....."
  },
  "orderInformation": {
    "amountDetails": {
      "authorizedAmount": "102.21",
      "currency": "USD"
    }
  },
  "paymentAccountInformation": {
    "card": {
      "type": "002"
    }
  },
  "paymentInformation": {
    "tokenizedCard": {
      "type": "002"
    },
    "instrumentIdentifier": {
      "id": "7020000000010603216",
      "state": "ACTIVE"
    },
    "shippingAddress": {
      "id": "F60328413BAE09A4E053AF598E0A33DB"
    },
```

```
    "paymentInstrument": {
      "id": "F6032841BE33098EE053AF598E0AB0A5"
    },
    "card": {
      "type": "002"
    },
    "customer": {
      "id": "F60328413BAB09A4E053AF598E0A33DB"
    }
  },
  "pointOfSaleInformation": {
    "terminalId": "08244117"
  },  "processingInformation": {    "paymentSolution": "014"  },
  "processorInformation": {
    "paymentAccountReferenceNumber": "50015OU4U5UYXLV127XTONYN49CL1",
    "merchantNumber": "000844028303882",
    "approvalCode": "831000",
    "networkTransactionId": "0602MCC603474",
    "transactionId": "0602MCC603474",
    "responseCode": "00",
    "avs": {
      "code": "Y",
      "codeRaw": "Y"
    }
  },
  "reconciliationId": "EUHW1EMHIZ3O",
  "status": "AUTHORIZED",
  "submitTimeUtc": "2023-03-03T17:31:48Z"
}
```

Making a Credit with a Customer Token

This section shows you how to make a credit with a customer token.

Endpoint

Test: `POST https://apitest.cybersource.com/pts/v2/credits`

Production: `POST https://api.cybersource.com/pts/v2/credits`

Production in India: `POST https://api.in.cybersource.com/pts/v2/credits`

Required Fields for Making a Credit with a Customer Token

clientReferenceInformation.code

orderInformation.amountDetails.currency

orderInformation.amountDetails.totalAmount

paymentInformation.customer.id

Set to the ID of the customer token you want to use.

Related Information

- API field reference guide for the REST API

REST Example: Making a Credit with a Customer Token

Request

```
{
  "clientReferenceInformation": {
    "code": "12345678"
  },
    "paymentInformation": {
        "customer": {
            "id": "F45FB3E443AC3C57E053A2598D0A9CFF"
        }
    },
    "orderInformation": {
```

```
        "amountDetails": {
            "currency": "USD",
            "totalAmount": "10.00"
        }
    }
}
```

Response to a Successful Request


```
{
  "_links": {
    "void": {
      "method": "POST",
      "href": "/pts/v2/credits/705596767782613290495 1/voids"
    },
    "self": {
      "method": "GET",
      "href": "/pts/v2/credits/70559676778261329049 51"
    }
  },
  "clientReferenceInformation": {
    "code": "12345678"
  },
  "creditAmountDetails": {
    "currency": "USD",
    "creditAmount": "10.00"
  },
  "id": "705596767782613290495 1",
  "orderInformation": {
    "amountDetails": {
      "currency": "USD"
    }
  },
  "paymentAccountInformation": {
    "card": {
      "type": "001"
    }
  },
  "paymentInformation": {
    "tokenizedCard": {
      "type": "001"
    },
    "instrumentIdentifier": {
      "id": "7030000000014831523",
      "state": "ACTIVE"
    },
    "shippingAddress": {
      "id": "F45FD8DE51B99E9CE053A2598D0AFDFA"
    },
    "paymentInstrument": {
      "id": "F45FE45E7993C7DBE053A2598D0AED19"
    },
    "card": {
      "type": "001"
    },
    "customer": {
```

```
      "id": "F45FB3E443AC3C57E053A2598D0A9CFF"
    }
  },
  "processorInformation": {
    "paymentAccountReferenceNumber": "V001001301932612153831309626 6",
    "approvalCode": "888888",
    "responseCode": "100"
  },
  "reconciliationId": "67444961BRIL0BB8",
  "status": "PENDING",
  "submitTimeUtc": "2024-01-18T16:52:48Z"
}
```

Shipping Address Tokens

The shipping address token contains the shipping address information associated with a customer token. This token includes any shipping address details, including the recipient's first and last name, company, shipping address, email, and phone number. A customer can have one or more shipping addresses, with one allocated as the customer's default shipping address.

Related information

Manage Shipping Address Tokens *(on page 126)*
Payments with Shipping Address Tokens *(on page 145)*

Manage Shipping Address Tokens

This section contains information managing shipping address tokens.

A shipping address token is associated with a customer token. You can create, retrieve, update, or delete an instrument identifier by submitting an HTTP `POST`, `GET`, `PATCH`, or `DELETE` operation to the `/tms/v2/customers/{customerTokenId}/shipping-addresses` endpoint. Use the TMS REST API shipping address endpoint to:

- Create a shipping address token (on page 126)
- Retrieve a shipping address token (on page 137) or multiple shipping address tokens for a specific customer (on page 139)
- Update a shipping address (on page 142)
- Delete a shipping address (on page 144)

For more information on shipping address tokens, see Shipping Address Tokens (on page 22).

Create a Customer Shipping Address

This section shows you how to create a customer shipping address.

Endpoint

Test: `POST https://apitest.cybersource.com/tms/v2/customers/{customerTokenId}/shipping-addresses`

Production: `POST https://api.cybersource.com/tms/v2/customers/{customerTokenId}/shipping-addresses`

Production in India: `POST https://api.in.cybersource.com/tms/v2/customers/{customerTokenId}/shipping-addresses`

The *{customerTokenId}* is the customer token ID returned in the **id** field when you created the customer token. For more information, see Create a Customer (on page 92).

If the default field is not supplied and the customer does not already have a shipping address, then the shipping address will become the default. Otherwise, it will become a customer's non-default shipping address.

Required Fields for Creating a Customer Shipping Address

You can include any of the following fields in the body of the request:

shipTo.address1

shipTo.address2

shipTo.administrativeArea

shipTo.company

shipTo.country

shipTo.email

shipTo.firstName

shipTo.lastName

shipTo.locality

shipTo.phoneNumber

shipTo.postalCode

Related Information

- API Field Reference for the REST API

REST Example: Creating a Customer Shipping Address

Request

```
POST https://apitest.cybersource.com/tms/v2/customers/F2F3ADA770102B51E053A2598D0A
9078/shipping-addresses
```

Response to a Successful Request


```
{
  "_links": {
    "self": {
      "href":
 "/tms/v2/customers/F2F3ADA770102B51E053A2598D0A9078/shipping-addresses/F2F4C2D1B9
66D631E053A2598D0AB155"
    },
    "customer": {
      "href": "/tms/v2/customers/F2F3ADA770102B51E053A2598D0A9078"
    }
  },
  "id": "F2F4C2D1B966D631E053A2598D0AB155",
  "default": true,
  "shipTo": {
    "firstName": "John",
    "lastName": "Doe",
    "company": "Company Name",
    "address1": "1 Market St",
    "locality": "San Francisco",
    "administrativeArea": "CA",
    "postalCode": "94105",
    "country": "US",
    "email": "test@cybs.com",
    "phoneNumber": "4158880000"
  },
  "metadata": {
    "creator": "testrest"
  }
}
```

Add a Default Shipping Address

This section shows you how to add a default customer shipping address.

Endpoint

Test: `POST https://apitest.cybersource.com/tms/v2/customers/{customerTokenId}/shipping-addresses`

Production: `POST https://api.cybersource.com/tms/v2/customers/{customerTokenId}/shipping-addresses`

Production in India: `POST https://api.in.cybersource.com/tms/v2/customers/{customerTokenId}/shipping-addresses`

The *{customerTokenId}* is the customer token ID returned in the **id** field when you created the customer token. For more information, see Create a Customer (on page 92).

Required Fields for Adding a Default Shipping Address

You can include any of the following fields in the body of the request:

default

Set to `true`.

shipTo.address1

shipTo.address2

shipTo.administrativeArea

shipTo.company

shipTo.country

shipTo.email

shipTo.firstName

shipTo.lastName

shipTo.locality

shipTo.phoneNumber

shipTo.postalCode

Related Information

- API Field Reference for the REST API

REST Example: Adding a Default Shipping Address

Request

```
{
    "default": true,
    "shipTo": {
        "firstName": "John",
        "lastName": "Doe",
        "company": "Visa",
        "address1": "1 Market St",
        "locality": "san francisco",
        "administrativeArea": "CA",
        "postalCode": "94105",
        "country": "US",
        "phoneNumber": "4158880000",
        "email": "test@cybs.com"
    }
}
```

Response to a Successful Request


```
{
    "_links": {
        "self": {
            "href":
 "/tms/v2/customers/F45FB3E443AC3C57E053A2598D0A9CFF/shipping-addresses/F45FD8DE51
A89E9CE053A2598D0AFDFA"
        },
        "customer": {
            "href": "/tms/v2/customers/F45FB3E443AC3C57E053A2598D0A9CFF"
        }
    },
    "id": "F45FD8DE51A89E9CE053A2598D0AFDFA",
    "default": true,
    "shipTo": {
        "firstName": "John",
        "lastName": "Doe",
        "company": "Visa",
        "address1": "1 Market St",
        "locality": "san francisco",
        "administrativeArea": "CA",
        "postalCode": "94105",
        "country": "US",
        "email": "test@cybs.com",
        "phoneNumber": "4158880000"
    },
    "metadata": {
        "creator": "testrest"
    }
}
```

Add a Non-Default Shipping Address

This section shows you how to add a non-default customer shipping address.

Endpoint

Test: `POST https://apitest.cybersource.com/tms/v2/customers/{customerTokenId}/shipping-addresses`

Production: `POST https://api.cybersource.com/tms/v2/customers/{customerTokenId}/shipping-addresses`

Production in India: `POST https://api.in.cybersource.com/tms/v2/customers/{customerTokenId}/shipping-addresses`

The *{customerTokenId}* is the customer token ID returned in the **id** field when you created the customer token. For more information, see Create a Customer Shipping Address (on page 126).

Required Fields for Adding a Non-Default Shipping Address

You can include any of the following fields in the body of the request:

default

Set to `false`.

shipTo.address1

shipTo.address2

shipTo.administrativeArea

shipTo.company

shipTo.country

shipTo.email

shipTo.firstName

shipTo.lastName

shipTo.locality

shipTo.phoneNumber

shipTo.postalCode

Related Information

- API Field Reference for the REST API

REST Example: Adding a Non-Default Shipping Address

Request

```
{
 "default": false,
        "shipTo": {
        "firstName": "John",
        "lastName": "Doe",
        "company": "Visa",
        "address1": "1 Market St",
        "locality": "san francisco",
        "administrativeArea": "CA",
        "postalCode": "94105",
        "country": "US",
        "phoneNumber": "4158880000",
        "email": "test@cybs.com"
    }
}
```

Response to a Successful Request


```
{
    "_links": {
        "self": {
            "href":
 "/tms/v2/customers/F45FB3E443AC3C57E053A2598D0A9CFF/shipping-addresses/F45FD8DE51
B99E9CE053A2598D0AFDFA"
        },
        "customer": {
            "href": "/tms/v2/customers/F45FB3E443AC3C57E053A2598D0A9CFF"
        }
    },
    "id": "F45FD8DE51B99E9CE053A2598D0AFDFA",
    "default": false,
    "shipTo": {
        "firstName": "John",
        "lastName": "Doe",
        "company": "Visa",
        "address1": "1 Market St",
        "locality": "san francisco",
        "administrativeArea": "CA",
        "postalCode": "94105",
        "country": "US",
        "email": "test@cybs.com",
        "phoneNumber": "4158880000"
    },
    "metadata": {
        "creator": "testrest"
    }
}
```

Change a Default Shipping Address

This section shows you how to change a default customer shipping address.

Endpoint

Test: `PATCH https://apitest.cybersource.com/tms/v2/customers/{customerTokenId}/shipping-addresses/{shippingAddressTokenId}`

Production: `PATCH https://api.cybersource.com/tms/v2/customers/{customerTokenId}/shipping-addresses/{shippingAddressTokenId}`

Production in India: `PATCH https://api.in.cybersource.com/tms/v2/customers/{customerTokenId}/shipping-addresses/{shippingAddressTokenId}`

The *{customerTokenId}* is the customer token ID returned in the **id** field when you created the customer token. In the *{shippingAddressTokenId}* path parameter, pass the shipping address token ID response field returned when you created a shipping address token. For more information, see Create a Customer (on page 92) and Create a Customer Shipping Address (on page 126).

Required Fields for Changing a Default Shipping Address

default

Set to `true`.

Related Information

- API Field Reference for the REST API

REST Example: Changing Default Shipping Address

Request

```
{
    "default": true
}
```

Response to a Successful Request


```
{
    "_links": {
        "self": {
            "href":
 "/tms/v2/customers/F45FB3E443AC3C57E053A2598D0A9CFF/shipping-addresses/F45FD8DE51
B99E9CE053A2598D0AFDFA"
        },
        "customer": {
            "href": "/tms/v2/customers/F45FB3E443AC3C57E053A2598D0A9CFF"
        }
    },
    "id": "F45FD8DE51B99E9CE053A2598D0AFDFA",
    "default": true,
    "shipTo": {
        "firstName": "John",
        "lastName": "Doe",
        "company": "Visa",
        "address1": "1 Market St",
        "locality": "san francisco",
        "administrativeArea": "CA",
        "postalCode": "94105",
        "country": "US",
        "email": "test@cybs.com",
        "phoneNumber": "4158880000"
    },
    "metadata": {
        "creator": "testrest"
    }
}
```


Retrieve a Customer Shipping Address

This section shows you how to retrieve a customer shipping address.

Endpoint

Test: `GET https://apitest.cybersource.com/tms/v2/customers/{customerTokenId}/shipping-addresses/{shippingAddressTokenId}`

Production: `GET https://api.cybersource.com/tms/v2/customers/{customerTokenId}/shipping-addresses/{shippingAddressTokenId}`

Production in India: `GET https://api.in.cybersource.com/tms/v2/customers/{customerTokenId}/shipping-addresses/{shippingAddressTokenId}`

The *{customerTokenId}* is the customer token ID returned in the **id** field when you created the customer token. In the *{shippingAddressTokenId}* path parameter, pass the shipping address token ID response field returned when you created a shipping address token. For more information, see Create a Customer (on page 92) and Create a Customer Shipping Address (on page 126).

REST Example: Retrieving a Shipping Address

Request

```
GET https://apitest.cybersource.com/tms/v2/customers/F2F3ADA770102B51E053A2598D0A9
078/shipping-addresses/F2F4C2D1B966D631E053A2598D0AB155
```

Response to a Successful Request

```
{
  "shipTo": {
    "firstName": "Jane",
    "lastName": "Smith",
    "company": "Lear Group, LLC",
    "address1": "123 Mountain Peaks Rd",
    "address2": "",
    "locality": "Mountain Peaks",
    "administrativeArea": "CA",
    "postalCode": "90212",
    "country": "US",
    "email": "jane.smith@leargroupllc.world",
    "phoneNumber": "123-456-7890"
  }
}
```

Retrieve All Customer Shipping Addresses

This section shows you how to retrieve all customer shipping addresses.

Endpoint

Test: `GET https://apitest.cybersource.com/tms/v2/customers/{customerTokenId}/shipping-addresses`

Production: `GET https://api.cybersource.com/tms/v2/customers/{customerTokenId}/shipping-addresses`

Production in India: `GET https://api.in.cybersource.com/tms/v2/customers/{customerTokenId}/shipping-addresses`

The *{customerTokenId}* is the customer token ID returned in the **id** field when you created the customer token. For more information, see Create a Customer (on page 92).

Use these query string parameters to filter the list of payment instrument tokens:

- `offset` — Page offset number.
- `limit` — Maximum number of items you would like returned.

REST Example: Retrieving All Customer Shipping Addresses

Request

```
GET https://apitest.cybersource.com/tms/v2/customers/F2F3ADA770102B51E053A2598D0A9
078/shipping-addresses?offset=0&limit=20
```

Response to a Successful Request

The shipping address in the first array element is the default shipping address.


```
{
  "_links": {
    "self": {
      "href":
 "/tms/v2/customers/F2F3ADA770102B51E053A2598D0A9078/shipping-addresses?offset=0&l
imit=20"
    },
    "first": {
      "href":
 "/tms/v2/customers/F2F3ADA770102B51E053A2598D0A9078/shipping-addresses?offset=0&l
imit=20"
    },
    "last": {
      "href":
 "/tms/v2/customers/F2F3ADA770102B51E053A2598D0A9078/shipping-addresses?offset=0&l
imit=20"
    }
  },
  "offset": 0,
  "limit": 20,
  "count": 1,
  "total": 1,
  "_embedded": {
    "shippingAddresses": [
      {
        "_links": {
          "self": {
            "href":
 "/tms/v2/customers/F2F3ADA770102B51E053A2598D0A9078/shipping-addresses/F2F4C2D1B9
66D631E053A2598D0AB155"
          },
          "customer": {
            "href": "/tms/v2/customers/F2F3ADA770102B51E053A2598D0A9078"
          }
        },
        "id": "F2F4C2D1B966D631E053A2598D0AB155",
        "default": true,
        "shipTo": {
          "firstName": "John",
          "lastName": "Doe",
          "company": "Company Name",
          "address1": "1 Market St",
          "locality": "San Francisco",
          "administrativeArea": "CA",
          "postalCode": "94105",
```

```
          "country": "US",
          "email": "test@cybs.com",
          "phoneNumber": "4158880000"
        },
        "metadata": {
          "creator": "testrest"
        }
      }
    ]
  }
}
```

Update a Customer Shipping Address

This section shows you how to update a customer shipping address.

Endpoint

Test: `PATCH https://apitest.cybersource.com/tms/v2/customers/{customerTokenId}/shipping-addresses/{shippingAddressTokenId}`

Production: `PATCH https://api.cybersource.com/tms/v2/customers/{customerTokenId}/shipping-addresses/{shippingAddressTokenId}`

Production in India: `PATCH https://api.in.cybersource.com/tms/v2/customers/{customerTokenId}/shipping-addresses/{shippingAddressTokenId}`

The *{customerTokenId}* is the customer token ID returned in the **id** field when you created the customer token. In the *{shippingAddressTokenId}* path parameter, pass the shipping address token ID response field returned when you created a shipping address token. For more information, see Create a Customer (on page 92) and Create a Customer Shipping Address (on page 126).

Required Fields for Updating a Customer Shipping Address

shipTo.address1

shipTo.address2

shipTo.administrativeArea

shipTo.company

shipTo.country

shipTo.email

shipTo.firstName

shipTo.lastName

shipTo.locality

shipTo.phoneNumber

shipTo.postalCode

Related Information

- API Field Reference for the REST API

REST Example: Updating a Customer Shipping Address

Request

```
{
  "shipTo": {
    "firstName": "Jane",
    "lastName": "Smith",
    "company": "Lear Group, LLC",
    "address1": "123 Mountain Peaks Rd",
    "address2": "Unit B",
    "locality": "Mountain Peaks",
    "administrativeArea": "CA",
    "postalCode": "90212",
    "country": "US",
    "email": "jane.smith@leargroupllc.world",
    "phoneNumber": "123-456-7890"
  }
}
```

Response to a Successful Request

```
{
  "shipTo": {
    "firstName": "Jane",
    "lastName": "Smith",
    "company": "Lear Group, LLC",
```

```
    "address1": "123 Mountain Peaks Rd",
    "address2": "Unit B",
    "locality": "Mountain Peaks",
    "administrativeArea": "CA",
    "postalCode": "90212",
    "country": "US",
    "email": "jane.smith@leargroupllc.world",
    "phoneNumber": "123-456-7890"
  }
}
```

Delete a Customer Shipping Address

This section shows you how to delete a customer shipping address.

Endpoint

Test: `DELETE https://apitest.cybersource.com/tms/v2/customers/{customerTokenId}/shipping-addresses/{shippingAddressTokenId}`

Production: `DELETE https://api.cybersource.com/tms/v2/customers/{customerTokenId}/shipping-addresses/{shippingAddressTokenId}`

Production in India: `DELETE https://api.in.cybersource.com/tms/v2/customers/{customerTokenId}/shipping-addresses/{shippingAddressTokenId}`

The *{customerTokenId}* is the customer token ID returned in the **id** field when you created the customer token. In the *{shippingAddressTokenId}* path parameter, pass the shipping address token ID response field returned when you created a shipping address token. For more information, see Create a Customer (on page 92) and Create a Customer Shipping Address (on page 126).

Important: If you have more than one shipping address, the default shipping address cannot be deleted without first selecting a new default shipping address.

Required Fields for Deleting a Customer Shipping Address

customerTokenId

Include the ID of the customer token you want to retrieve in the URL path.

shippingAddressTokenId

Include the ID of the shipping address token you want to retrieve in the URL path.

Related Information

- API Field Reference for the REST API

REST Example: Deleting a Customer Shipping Address

Request

```
DELETE https://apitest.cybersource.com/tms/v2/customers/F2F3ADA770102B51E053A2598D
0A9078/shipping-addresses/F2F4C2D1B966D631E053A2598D0AB155
```

Response to a Successful Request

A successful delete response returns an empty `HTTP 204 No Content` status. For more information, see HTTP Status Codes (on page 358).

Payments with Shipping Address Tokens

This section contains information on making payments with shipping address tokens.

A shipping address token is associated with a specific customer token. This includes any shipping address details, including first and last name, company, shipping address, email, and phone number.

To make a payment using a shipping address token, you must either create the token in the authorization request or include the customer token ID as the value in the `paymentInformation.customer.id` and `paymentInformation.shippingAddress.id` fields. You can make payments using non-default shipping address tokens. For example:

- Authorizing a Payment with a Non-Default Shipping Address (on page 146)

For more information on shipping address tokens, see Shipping Address Tokens (on page 22).

Authorizing a Payment with a Non-Default Shipping Address

This section provides the information you need in order to make a payment with a non-default shipping address.

Endpoint

Production: `POST https://api.cybersource.com/pts/v2/payments`

Test: `POST https://apitest.cybersource.com/pts/v2/payments`

Production in India: `POST https://api.in.cybersource.com/pts/v2/payments`

Required Fields for Authorizing a Payment with a Non-Default Shipping Address

clientReferenceInformation.code

orderInformation.amountDetails.currency

orderInformation.amountDetails.totalAmount

paymentInformation.customer.id

Set to the ID of the customer token you want to use.

paymentInformation.shippingAddress.id

Set to the ID of the shipping address token you want to use.

Related Information

- API field reference guide for the REST API

REST Example: Authorizing a Payment with a Non-Default Shipping Address

Request

```
{
  "clientReferenceInformation": {
    "code": "12345678"
  },
```

```
    "paymentInformation": {
        "customer": {
            "id": "F45FB3E443AC3C57E053A2598D0A9CFF"
        },
        "shippingAddress": {
            "id": "F45FD8DE51B99E9CE053A2598D0AFDFA"
        }
    },
    "orderInformation": {
        "amountDetails": {
            "currency": "USD",
            "totalAmount": "10.00"
        }
    }
}
```

Response to a Successful Request


```
{
  "_links": {
    "authReversal": {
      "method": "POST",
      "href": "/pts/v2/payments/705594903731678690 4953/reversals"
    },
    "self": {
      "method": "GET",
      "href": "/pts/v2/payments/705594903731678690 4953"
    },
    "capture": {
      "method": "POST",
      "href": "/pts/v2/payments/705594903731678690 4953/captures"
    }
  },
  "clientReferenceInformation": {
    "code": "12345678"
  },
  "id": "7055949037316786904953",
  "orderInformation": {
    "amountDetails": {
      "authorizedAmount": "10.00",
      "currency": "USD"
    }
  },
  "paymentAccountInformation": {
    "card": {
      "type": "001"
    }
  },
  "paymentInformation": {
    "tokenizedCard": {
      "type": "001"
    },
    "instrumentIdentifier": {
      "id": "7030000000014831523",
      "state": "ACTIVE"
    },
    "shippingAddress": {
      "id": "F45FD8DE51B99E9CE053A2598D0AFDFA"
    },
    "paymentInstrument": {
      "id": "F45FE45E7993C7DBE053A2598D0AED19"
    },
    "card": {
      "type": "001"
    },
```

```
    "customer": {
      "id": "F45FB3E443AC3C57E053A2598D0A9CFF"
    }
  },
  "pointOfSaleInformation": {
    "terminalId": "111111"
  },
  "processorInformation": {
    "approvalCode": "888888",
    "networkTransactionId": "123456789619999",
    "transactionId": "123456789619999",
    "responseCode": "100",
    "avs": {
      "code": "X",
      "codeRaw": "I1"
    }
  },
  "reconciliationId": "674679208RIKQ52K",
  "status": "AUTHORIZED",
  "submitTimeUtc": "2024-01-18T16:21:44Z"
}
```

Customer Payment Instrument Tokens

Customer payment instruments are payment instruments that are linked to a specific customer token. Supported payment instruments include payment cards, tokenized cards (Apple Pay and Android Pay), or ACH bank accounts.

Manage Customer Payment Instrument Tokens

This section contains information on managing customer payment instrument tokens.

Customer payment instruments are payment instruments that are linked to a specific customer token. The following payment instruments are supported:

- Payment card
- Tokenized card (Apple Pay and Android Pay)
- ACH bank account

You can create, retrieve, update, or delete a payment instrument by submitting an HTTP `POST`, `GET`, `PATCH`, or `DELETE` operation to the `tms/v2/customers/{customerTokenId}/payment-instruments` endpoint. Use the TMS REST API payment instrument endpoint to:

- Create a customer payment instrument token (on page 150)
- Retrieve a customer payment instrument token (on page 178)
- Update a customer payment instrument (on page 190)
- Delete a customer payment instrument (on page 195)

For more information on customer tokens and payment instrument tokens, see Customer Tokens (on page 22) and Payment Instrument Tokens (on page 22), respectively.

Create a Customer Payment Instrument

This section shows you how to create a customer payment instrument token.

Endpoint

Test: `POST https://apitest.cybersource.com/tms/v2/customers/{customerTokenId}/payment-instruments`

Production: `POST https://api.cybersource.com/tms/v2/customers/{customerTokenId}/payment-instruments`

Production in India: `POST https://api.in.cybersource.com/tms/v2/customers/{customerTokenId}/payment-instruments`

The *{customerTokenId}* is the customer token ID returned in the **id** field when you created the customer token. For more information, see Create a Customer (on page 92).

Required Fields for Creating a Customer Payment Instrument

card.type

Required if the instrument identifier ID being linked to is card-based.

Related Information

- API Field Reference for the REST API

Optional Fields for Creating a Customer Payment Instrument

bankAccount.type

billTo.address1

billTo.address2

billTo.aminstrativeArea

billTo.company

billTo.country

billTo.email

billTo.firstName

billTo.lastName

billTo.locality

billTo.phoneNumber

billTo.postalCode

buyerInformation.companyTaxID

buyerInformation.currency

buyerInformation.dateOfBirth

buyerInformation.personalIdentification.id

buyerInformation.personalIdentification.issuedBy.administrativeArea

buyerInformation.personalIdentification.type

card.expirationMonth

card.expirationYear

card.issueNumber

card.startMonth

card.startYear

card.useAs

card.tokenizedInformation.requestorID

card.tokenizedInformation.transactionType

default

If you do not include this field, the first payment instrument for a customer becomes the default. A subsequent payment instrument becomes the non-default option.

instrumentIdentifier.id

processingInformation.billPaymentProgramEnabled

merchantInformation.merchantDescriptor.alternateName

Related Information

- API Field Reference for the REST API

REST Example: Creating a Customer Payment Instrument

Request

```
{
  "card": {
    "expirationMonth": "12",
    "expirationYear": "2031",
    "type": "001"
  },
  "billTo": {
    "firstName": "John",
    "lastName": "Doe",
    "company": "Company Name",
    "address1": "1 Market St",
    "locality": "San Francisco",
    "administrativeArea": "CA",
    "postalCode": "94105",
    "country": "US",
    "email": "test@cybs.com",
    "phoneNumber": "4158880000"
  },
  "instrumentIdentifier": {
    "id": "7010000000016241111"
  }
}
```

Response to a Successful Request


```
{
  "_links": {
    "self": {
      "href":
 "/tms/v2/customers/F2F3ADA770102B51E053A2598D0A9078/payment-instruments/F39732BE4
BDA9A1EE053AF598E0A4081"
    },
    "customer": {
      "href": "/tms/v2/customers/F2F3ADA770102B51E053A2598D0A9078"
    }
  },
  "id": "F39732BE4BDA9A1EE053AF598E0A4081",
  "default": true,
  "state": "ACTIVE",
  "card": {
    "expirationMonth": "12",
    "expirationYear": "2031",
    "type": "001"
  },
  "billTo": {
    "firstName": "John",
    "lastName": "Doe",
    "company": "Company Name",
    "address1": "1 Market St",
    "locality": "San Francisco",
    "administrativeArea": "CA",
    "postalCode": "94105",
    "country": "US",
    "email": "test@cybs.com",
    "phoneNumber": "4158880000"
  },
  "instrumentIdentifier": {
    "id": "7010000000016241111"
  },
  "metadata": {
    "creator": "testrest"
  },
  "_embedded": {
    "instrumentIdentifier": {
      "_links": {
        "self": {
          "href": "/tms/v1/instrumentidentifiers/7010000000016241111"
        },
        "paymentInstruments": {
          "href":
 "/tms/v1/instrumentidentifiers/7010000000016241111/paymentinstruments"
        }
```

```
      },
      "id": "7010000000016241111",
      "object": "instrumentIdentifier",
      "state": "ACTIVE",
      "card": {
        "number": "411111XXXXXX1111"
      },
      "processingInformation": {
        "authorizationOptions": {
          "initiator": {
            "merchantInitiatedTransaction": {
              "previousTransactionId": "123456789012345"
            }
          }
        }
      },
      "metadata": {
        "creator": "testrest"
      }
    }
  }
}
}
```

Add a Default Payment Instrument Using Instrument Identifier

This section shows you how add a default payment instrument using an instrument identifier.

Endpoint

Test: `POST https://apitest.cybersource.com/tms/v2/customers/{customerTokenId}/payment-instruments`

Production: `POST https://api.cybersource.com/tms/v2/customers/{customerTokenId}/payment-instruments`

Production in India: `POST https://api.in.cybersource.com/tms/v2/customers/{customerTokenId}/payment-instruments`

The *{customerTokenId}* is the customer token ID returned in the **id** field when you created the customer token. For more information, see Create a Customer (on page 92).

Required Fields for Adding a Default Payment Instrument Using Instrument Identifier Using the REST API

default

Set value to `true`.

Related Information

- API Field Reference for the REST API

Optional Fields for Adding a Default Payment Instrument Using Instrument Identifier Using the REST API

bankAccount.type

billTo.address1

billTo.address2

billTo.aminstrativeArea

billTo.company

billTo.country

billTo.email

billTo.firstName

billTo.lastName

billTo.locality

billTo.phoneNumber

billTo.postalCode

buyerInformation.companyTaxID

buyerInformation.currency

buyerInformation.dateOfBirth

buyerInformation.personalIdentification.id

buyerInformation.personalIdentification.issuedBy.administrativeArea

buyerInformation.personalIdentification.type

card.expirationMonth

card.expirationYear

card.issueNumber

card.startMonth

card.startYear

card.type

card.useAs

card.tokenizedInformation.requestorID

card.tokenizedInformation.transactionType

instrumentIdentifier.id

processingInformation.billPaymentProgramEnabled

merchantInformation.merchantDescriptor.alternateName

Related Information

- API Field Reference for the REST API

REST Example: Adding a Default Payment Instrument Using Instrument Identifier

Request

```
{
    "default": true,
    "card": {
        "expirationMonth": "12",
        "expirationYear": "2031",
        "type": "001"
    },
    "billTo": {
        "firstName": "John",
        "lastName": "Doe",
        "company": "Visa",
        "address1": "1 Market St",
        "locality": "san francisco",
        "administrativeArea": "CA",
        "postalCode": "94105",
        "country": "US",
        "phoneNumber": "4158880000",
        "email": "test@cybs.com"
    },
    "instrumentIdentifier": {
        "id": "7010000000016241111"
    }
}
```

Response to a Successful Request


```
{
    "_links": {
        "self": {
            "href":
 "/tms/v2/customers/F45FB3E443AC3C57E053A2598D0A9CFF/payment-instruments/F45FD8DE5
42A9E9CE053A2598D0AFDFA"
        },
        "customer": {
            "href": "/tms/v2/customers/F45FB3E443AC3C57E053A2598D0A9CFF"
        }
    },
    "id": "F45FD8DE542A9E9CE053A2598D0AFDFA",
    "default": true,
    "state": "ACTIVE",
    "card": {
        "expirationMonth": "12",
        "expirationYear": "2031",
        "type": "001"
    },
    "billTo": {
        "firstName": "John",
        "lastName": "Doe",
        "company": "Visa",
        "address1": "1 Market St",
        "locality": "san francisco",
        "administrativeArea": "CA",
        "postalCode": "94105",
        "country": "US",
        "email": "test@cybs.com",
        "phoneNumber": "4158880000"
    },
    "instrumentIdentifier": {
        "id": "7030000000014911515"
    },
    "metadata": {
        "creator": "testrest"
    },
    "_embedded": {
        "instrumentIdentifier": {
            "_links": {
                "self": {
                    "href": "/tms/v1/instrumentidentifiers/7030000000014911515"
                },
                "paymentInstruments": {
                    "href":
 "/tms/v1/instrumentidentifiers/7030000000014911515/paymentinstruments"
                }
```

```
            },
            "id": "7030000000014911515",
            "object": "instrumentIdentifier",
            "state": "ACTIVE",
            "card": {
                "number": "489537XXXXXX1515"
            },
            "issuer": {
                "paymentAccountReference": "V0010013019326121174070050420"
            },
            "processingInformation": {
                "authorizationOptions": {
                    "initiator": {
                        "merchantInitiatedTransaction": {
                            "previousTransactionId": "123456789619999"
                        }
                    }
                }
            },
            "metadata": {
                "creator": "testrest"
            }
        }
    }
}
```

Add a Default Payment Instrument with Validated Payment

This section shows you how to add a default payment instrument with a validated payment method.

Endpoint

Test: `POST https://apitest.cybersource.com/pts/v2/payments`

Production: `POST https://api.cybersource.com/pts/v2/payments`

Production in India: `POST https://api.in.cybersource.com/pts/v2/payments`

Required Fields for Adding a Default Payment Instrument with Validated Payment Using the REST API

orderInformation.amountDetails.currency

orderInformation.amountDetails.totalAmount

orderInformation.billTo.address1

orderInformation.billTo.administrativeArea

orderInformation.billTo.country

orderInformation.billTo.email

orderInformation.billTo.firstName

orderInformation.billTo.lastName

orderInformation.billTo.locality

orderInformation.billTo.postalCode

paymentInformation.card.expirationMonth

paymentInformation.card.expirationYear

paymentInformation.card.number

paymentInformation.card.type

paymentInformation.customer.id

Set the value to the ID of the customer token.

processingInformation.actionList

Set the value to `TOKEN_CREATE`.

processingInformation.actionTokenTypes

Set the value to `paymentInstrument`.

tokenInformation.paymentInstrument.default

Set value to `true`.

Related Information

- API Field Reference for the REST API

REST Example: Adding a Default Payment Instrument with Validated Payment

Request

```
{
    "clientReferenceInformation": {
        "code": "TC50171_3"
    },
    "processingInformation": {
        "commerceIndicator": "internet",
        "actionList": [
            "TOKEN_CREATE"
        ],
        "actionTokenTypes": [
            "paymentInstrument"
        ]
    },
    "orderInformation": {
        "billTo": {
            "country": "US",
            "lastName": "Deo",
            "address2": "Address 2",
            "address1": "201 S. Division St.",
            "postalCode": "48104-2201",
            "locality": "Ann Arbor",
            "administrativeArea": "MI",
            "firstName": "John",
            "phoneNumber": "999999999",
            "district": "MI",
            "buildingNumber": "123",
            "company": "Visa",
            "email": "test@cybs.com"
        },
```

```
        "shipTo": {
            "country": "US",
            "lastName": "Deo",
            "address2": "Address 2",
            "address1": "201 S. Division St.",
            "postalCode": "48104-2201",
            "locality": "Ann Arbor",
            "administrativeArea": "MI",
            "firstName": "John",
            "phoneNumber": "999999999",
            "district": "MI",
            "buildingNumber": "123",
            "company": "Visa",
            "email": "test@cybs.com"
        },
        "amountDetails": {
            "totalAmount": "102.00",
            "currency": "USD"
        }
    },
    "paymentInformation": {
        "customer": {
            "id": "{{tms-customer-id}}"
        },
        "card": {
            "expirationYear": "2031",
            "number": "4895379987X11523",
            "securityCode": "965",
            "expirationMonth": "12",
            "type": "001"
        }
    },
    "tokenInformation": {
        "paymentInstrument": {
            "default": "true"
        }
    }
}
```

Response to a Successful Request


```
{
    "_links": {
        "authReversal": {
            "method": "POST",
            "href": "/pts/v2/payments/676063774731617320395 5/reversals"
        },
        "self": {
            "method": "GET",
            "href": "/pts/v2/payments/67606377473161732039 55"
        },
        "capture": {
            "method": "POST",
            "href": "/pts/v2/payments/676063774731617320395 5/captures"
        }
    },
    "clientReferenceInformation": {
        "code": "TC50171_3"
    },
    "id": "6760637747316173203955",
    "orderInformation": {
        "amountDetails": {
            "authorizedAmount": "102.00",
            "currency": "USD"
        }
    },
    "paymentAccountInformation": {
        "card": {
            "type": "001"
        }
    },
    "paymentInformation": {
        "tokenizedCard": {
            "type": "001"
        },
        "shippingAddress": {
            "id": "F45FD8DE51B99E9CE053A2598D0AFDFA"
        },
        "card": {
            "type": "001"
        },
        "customer": {
            "id": "F45FB3E443AC3C57E053A2598D0A9CFF"
        }
    },
    "pointOfSaleInformation": {
        "terminalId": "111111"
    },
```

```
    "processorInformation": {
        "paymentAccountReferenceNumber": "V0010013019326121538313096266",
        "approvalCode": "888888",
        "networkTransactionId": "123456789619999",
        "transactionId": "123456789619999",
        "responseCode": "100",
        "avs": {
            "code": "X",
            "codeRaw": "I1"
        }
    },
    "reconciliationId": "69815876LDTHD4XU",
    "status": "AUTHORIZED",
    "submitTimeUtc": "2023-02-10T21:16:15Z",
    "tokenInformation": {
        "instrumentidentifierNew": false,
        "instrumentIdentifier": {
            "state": "ACTIVE",
            "id": "7030000000014831523"
        },
        "paymentInstrument": {
            "id": "F45FE45E7993C7DBE053A2598D0AED19"
        }
    }
}
```

Add a Non-Default Payment Instrument Using Instrument Identifier

This section shows you how to add a non-default payment instrument using instrument identifier.

Endpoint

Test: `POST https://apitest.cybersource.com/tms/v2/customers/{customerTokenId}/payment-instruments/{paymentInstrumentTokenId}`

Production: `POST https://api.cybersource.com/tms/v2/customers/{customerTokenId}/payment-instruments/{paymentInstrumentTokenId}`

Production in India: `POST https://api.in.cybersource.com/tms/v2/customers/{customerTokenId}/payment-instruments/{paymentInstrumentTokenId}`

The *{customerTokenId}* is the customer token ID returned in the **id** field when you created the customer token. The *{paymentInstrumentTokenId}* is the payment instrument token ID you want to retrieve. For more information, see Create a Customer (on page 92) and Create a Customer Payment Instrument (on page 150).

Required Fields for Adding a Non-Default Payment Instrument Using Instrument Identifier

customerTokenId

Include the ID of the customer token you want to retrieve in the URL path.

paymentInstrumentTokenId

Include the ID of the payment instrument token you want to retrieve in the URL path.

Related Information

- API Field Reference for the REST API

Optional Fields for Adding a Non-Default Payment Instrument Using Instrument Identifier

bankAccount.type

billTo.address1

billTo.address2

billTo.aminstrativeArea

billTo.company

billTo.country

billTo.email

billTo.firstName

billTo.lastName

billTo.locality

billTo.phoneNumber

billTo.postalCode

buyerInformation.companyTaxID

buyerInformation.currency

buyerInformation.dateOfBirth

buyerInformation.personalIdentification.id

buyerInformation.personalIdentification.issuedBy.administrativeArea

buyerInformation.personalIdentification.type

card.expirationMonth

card.expirationYear

card.issueNumber

card.startMonth

card.startYear

card.type

card.useAs

card.tokenizedInformation.requestorID

card.tokenizedInformation.transactionType

default

Set value to `true` if default, otherwise set value to `false`.

instrumentIdentifier.id

Set the value to the ID of the instrument identifier token.

processingInformation.billPaymentProgramEnabled

merchantInformation.merchantDescriptor.alternateName

Related Information

- API Field Reference for the REST API

REST Example: Adding a Non-Default Payment Instrument Using Instrument Identifier

Request

```
{
    "default": false,
    "card": {
        "expirationMonth": "12",
        "expirationYear": "2031",
        "type": "001"
    },
    "billTo": {
        "firstName": "John",
        "lastName": "Doe",
        "company": "Visa",
        "address1": "1 Market St",
        "locality": "san francisco",
        "administrativeArea": "CA",
        "postalCode": "94105",
        "country": "US",
        "phoneNumber": "4158880000",
        "email": "test@cybs.com"
    },
    "instrumentIdentifier": {
        "id": "{{tms-instrumentIdentifier-id}}"
    }
}
```

Response to a Successful Request


```
{
    "_links": {
        "self": {
            "href":
 "/tms/v2/customers/F45FB3E443AC3C57E053A2598D0A9CFF/payment-instruments/F45FE3A5D
AD6CF8CE053A2598D0AA1EF"
        },
        "customer": {
            "href": "/tms/v2/customers/F45FB3E443AC3C57E053A2598D0A9CFF"
        }
    },
    "id": "F45FE3A5DAD6CF8CE053A2598D0AA1EF",
    "default": false,
    "state": "ACTIVE",
    "card": {
        "expirationMonth": "12",
        "expirationYear": "2031",
        "type": "001"
    },
    "billTo": {
        "firstName": "John",
        "lastName": "Doe",
        "company": "Visa",
        "address1": "1 Market St",
        "locality": "san francisco",
        "administrativeArea": "CA",
        "postalCode": "94105",
        "country": "US",
        "email": "test@cybs.com",
        "phoneNumber": "4158880000"
    },
    "instrumentIdentifier": {
        "id": "7030000000012931531"
    },
    "metadata": {
        "creator": "testrest"
    },
    "_embedded": {
        "instrumentIdentifier": {
            "_links": {
                "self": {
                    "href": "/tms/v1/instrumentidentifiers/7030000000012931531"
                },
                "paymentInstruments": {
                    "href":
 "/tms/v1/instrumentidentifiers/7030000000012931531/paymentinstruments"
                }
```

```
            },
            "id": "7030000000012931531",
            "object": "instrumentIdentifier",
            "state": "ACTIVE",
            "card": {
                "number": "489537XXXXXX1531"
            },
            "issuer": {
                "paymentAccountReference": "V0010013019326121921451482293"
            },
            "processingInformation": {
                "authorizationOptions": {
                    "initiator": {
                        "merchantInitiatedTransaction": {
                            "previousTransactionId": "123456789619999"
                        }
                    }
                }
            },
            "metadata": {
                "creator": "testrest"
            }
        }
    }
}
```

Add a Non-Default Payment Instrument with Validated Payment

This section shows you how to add a non-default payment instrument with a validated payment.

Endpoint

Test: `GET https://apitest.cybersource.com/pts/v2/payments`

Production: `GET https://api.cybersource.com/pts/v2/payments`

Production in India: `GET https://api.in.cybersource.com/pts/v2/payments`

The *`{customerTokenId}`* is the customer token ID returned in the **id** field when you created the customer token. For more information, see Create a Customer (on page 92).

Required Fields for Adding a Non-Default Payment Instrument with Validated Payment Using the REST API

orderInformation.amountDetails.currency

orderInformation.amountDetails.totalAmount

orderInformation.billTo.address1

orderInformation.billTo.administrativeArea

orderInformation.billTo.country

orderInformation.billTo.email

orderInformation.billTo.firstName

orderInformation.billTo.lastName

orderInformation.billTo.locality

orderInformation.billTo.postalCode

paymentInformation.card.expirationMonth

paymentInformation.card.expirationYear

paymentInformation.card.number

paymentInformation.card.type

paymentInformation.customer.id

Set the value to the ID of the customer token.

processingInformation.actionList

Set the value to `TOKEN_CREATE`.

processingInformation.actionTokenTypes

Set the value to `paymentInstrument`.

tokenInformation.paymentInstrument.default

Set value to `false`.

Related Information

- API Field Reference for the REST API

REST Example: Adding a Non-Default Payment Instrument with Validated Payment

Request

```
{
    "clientReferenceInformation": {
        "code": "TC50171_3"
    },
    "processingInformation": {
        "commerceIndicator": "internet",
        "actionList": [
            "TOKEN_CREATE"
        ],
        "actionTokenTypes": [
            "paymentInstrument"
        ]
    },
    "orderInformation": {
        "billTo": {
            "country": "US",
            "lastName": "Deo",
            "address2": "Address 2",
            "address1": "201 S. Division St.",
            "postalCode": "48104-2201",
            "locality": "Ann Arbor",
            "administrativeArea": "MI",
            "firstName": "John",
```

```
            "phoneNumber": "999999999",
            "district": "MI",
            "buildingNumber": "123",
            "company": "Visa",
            "email": "test@cybs.com"
        },
        "shipTo": {
            "country": "US",
            "lastName": "Deo",
            "address2": "Address 2",
            "address1": "201 S. Division St.",
            "postalCode": "48104-2201",
            "locality": "Ann Arbor",
            "administrativeArea": "MI",
            "firstName": "John",
            "phoneNumber": "999999999",
            "district": "MI",
            "buildingNumber": "123",
            "company": "Visa",
            "email": "test@cybs.com"
        },
        "amountDetails": {
            "totalAmount": "102.00",
            "currency": "USD"
        }
    },
    "paymentInformation": {
        "customer": {
            "id": "{{tms-customer-id}}"
        },
        "card": {
            "expirationYear": "2031",
            "number": "4895379987X11531",
            "securityCode": "258",
            "expirationMonth": "12",
            "type": "001"
        }
    },
    "tokenInformation": {
        "paymentInstrument": {
            "default": "false"
        }
    }
}
```

Response to a Successful Request


```
{
    "_links": {
        "authReversal": {
            "method": "POST",
            "href": "/pts/v2/payments/676063808431617570395 5/reversals"
        },
        "self": {
            "method": "GET",
            "href": "/pts/v2/payments/6760638084316175703955"
        },
        "capture": {
            "method": "POST",
            "href": "/pts/v2/payments/676063808431617570395 5/captures"
        }
    },
    "clientReferenceInformation": {
        "code": "TC50171_3"
    },
    "id": "6760638084316175703955",
    "orderInformation": {
        "amountDetails": {
            "authorizedAmount": "102.00",
            "currency": "USD"
        }
    },
    "paymentAccountInformation": {
        "card": {
            "type": "001"
        }
    },
    "paymentInformation": {
        "tokenizedCard": {
            "type": "001"
        },
        "shippingAddress": {
            "id": "F45FD8DE51B99E9CE053A2598D0AFDFA"
        },
        "card": {
            "type": "001"
        },
        "customer": {
            "id": "F45FB3E443AC3C57E053A2598D0A9CFF"
        }
    },
    "pointOfSaleInformation": {
        "terminalId": "111111"
    },
```

```
    "processorInformation": {
        "paymentAccountReferenceNumber": "V0010013019326121921451482293",
        "approvalCode": "888888",
        "networkTransactionId": "123456789619999",
        "transactionId": "123456789619999",
        "responseCode": "100",
        "avs": {
            "code": "X",
            "codeRaw": "I1"
        }
    },
    "reconciliationId": "698162754DTIATRS",
    "status": "AUTHORIZED",
    "submitTimeUtc": "2023-02-10T21:16:48Z",
    "tokenInformation": {
        "instrumentidentifierNew": false,
        "instrumentIdentifier": {
            "state": "ACTIVE",
            "id": "7030000000012931531"
        },
        "paymentInstrument": {
            "id": "F45FE45E79DCC7DBE053A2598D0AED19"
        }
    }
}
```

Change a Customer's Default Payment Instrument

This section shows you how to change a customer's default payment instrument.

Endpoint

Test: `PATCH https://apitest.cybersource.com/tms/v2/customers/{customerTokenId}/payment-instruments/{paymentInstrumentTokenId}`

Production: `PATCH https://api.cybersource.com/tms/v2/customers/{customerTokenId}/payment-instruments/{paymentInstrumentTokenId}`

Production in India: `PATCH https://api.in.cybersource.com/tms/v2/customers/{customerTokenId}/payment-instruments/{paymentInstrumentTokenId}`

The *{customerTokenId}* is the customer token ID returned in the **id** field when you created the customer token. The *{paymentInstrumentTokenId}* is the payment instrument token ID you want to retrieve. For more information, see Create a Customer (on page 92) and Create a Customer Payment Instrument (on page 150).

Required Fields for Changing a Customer's Default Payment Instrument

default

Set value to `true`.

Related Information

- API Field Reference for the REST API

REST Example: Changing a Customer's Default Payment Instrument

Request

```
{
    "default": true
}
```

Response to a Successful Request


```
{
    "_links": {
        "self": {
            "href":
 "/tms/v2/customers/F4D5E715F75E9910E053A2598D0A7278/payment-instruments/F4D5E715F
7BD9910E053A2598D0A7278"
        },
        "customer": {
            "href": "/tms/v2/customers/F4D5E715F75E9910E053A2598D0A7278"
        }
    },
    "id": "F4D5E715F7BD9910E053A2598D0A7278",
    "default": true,
    "state": "ACTIVE",
    "card": {
        "expirationMonth": "12",
        "expirationYear": "2031",
        "type": "001"
    },
    "billTo": {
        "firstName": "John",
        "lastName": "Doe",
        "company": "Visa",
        "address1": "1 Market St",
        "locality": "san francisco",
        "administrativeArea": "CA",
        "postalCode": "94105",
        "country": "US",
        "email": "test@cybs.com",
        "phoneNumber": "4158880000"
    },
    "instrumentIdentifier": {
        "id": "7010000000016241111"
    },
    "metadata": {
        "creator": "testrest"
    },
    "_embedded": {
        "instrumentIdentifier": {
            "_links": {
                "self": {
                    "href": "/tms/v1/instrumentidentifiers/7010000000016241111"
                },
                "paymentInstruments": {
                    "href":
 "/tms/v1/instrumentidentifiers/7010000000016241111/paymentinstruments"
                }
```

```
            },
            "id": "7010000000016241111",
            "object": "instrumentIdentifier",
            "state": "ACTIVE",
            "card": {
                "number": "411111XXXXXX1111"
            },
            "processingInformation": {
                "authorizationOptions": {
                    "initiator": {
                        "merchantInitiatedTransaction": {
                            "previousTransactionId": "123456789619999"
                        }
                    }
                }
            },
            "metadata": {
                "creator": "testrest"
            }
        }
    }
}
```

Retrieve a Customer Payment Instrument

This section shows you how to retrieve a customer payment instrument token.

Endpoint

Test: `GET https://apitest.cybersource.com/tms/v2/customers/{customerTokenId}/payment-instruments/{paymentInstrumentTokenId}`

Production: `GET https://api.cybersource.com/tms/v2/customers/{customerTokenId}/payment-instruments/{paymentInstrumentTokenId}`

Production in India: `GET https://api.in.cybersource.com/tms/v2/customers/{customerTokenId}/payment-instruments/{paymentInstrumentTokenId}`

The *{customerTokenId}* is the customer token ID returned in the **id** field when you created the customer token. The *{paymentInstrumentTokenId}* is the payment instrument token ID you want to retrieve. For more information, see Create a Customer (on page 92) and Create a Customer Payment Instrument (on page 150).

REST Example: Retrieving a Customer Payment Instrument

Request

```
GET https://apitest.cybersource.com/tms/v2/customers/F2F3ADA770102B51E053A2598D0A9
078/payment-instruments/F39732BE4BDA9A1EE053AF598E0A4081
```

Response to a Successful Request


```
{
  "_links": {
    "self": {
      "href":
 "/tms/v2/customers/F2F3ADA770102B51E053A2598D0A9078/payment-instruments/F39732BE4
BDA9A1EE053AF598E0A4081"
    },
    "customer": {
      "href": "/tms/v2/customers/F2F3ADA770102B51E053A2598D0A9078"
    }
  },
  "id": "F39732BE4BDA9A1EE053AF598E0A4081",
  "default": true,
  "state": "ACTIVE",
  "card": {
    "expirationMonth": "12",
    "expirationYear": "2031",
    "type": "001"
  },
  "billTo": {
    "firstName": "John",
    "lastName": "Doe",
    "company": "Company Name",
    "address1": "1 Market St",
    "locality": "San Francisco",
    "administrativeArea": "CA",
    "postalCode": "94105",
    "country": "US",
    "email": "test@cybs.com",
    "phoneNumber": "4158880000"
  },
  "instrumentIdentifier": {
    "id": "7010000000016241111"
  },
  "metadata": {
    "creator": "testrest"
  },
  "_embedded": {
    "instrumentIdentifier": {
      "_links": {
        "self": {
          "href": "/tms/v1/instrumentidentifiers/7010000000016241111"
        },
        "paymentInstruments": {
          "href":
 "/tms/v1/instrumentidentifiers/7010000000016241111/paymentinstruments"
        }
```

```
      },
      "id": "7010000000016241111",
      "object": "instrumentIdentifier",
      "state": "ACTIVE",
      "card": {
        "number": "411111XXXXXX1111"
      },
      "processingInformation": {
        "authorizationOptions": {
          "initiator": {
            "merchantInitiatedTransaction": {
              "previousTransactionId": "123456789012345"
            }
          }
        }
      },
      "metadata": {
        "creator": "testrest"
      }
    }
  }
}
```

Retrieve a Customer Payment Instrument with an Unmasked Card Number

This section shows you how to retrieve a payment instrument with an unmasked card number.

Important: To retrieve unmasked payment details, you must ensure that your MLE key pair and your token vault are configured correctly. For more information on MLE keys, see Message-Level Encryption Keys (on page 65). For more information on token vaults, see Token Vault Management (on page 62). If necessary, contact your Cybersource account manager or customer support.

The response is BASE 64-encoded JSON web encryption (JWE) token. The decoded JWE has these elements:

```
{ "alg": "RSA-OAEP-256", //The algorithm used to encrypt the CEK
    "cty": "json", //The content type
    "typ": "JWT", //The token type
    "enc": "A256GCM", //The algorithm that is used to encrypt the message
    "kid": "keyId" //The serial number of shared public cert for encryption of CEK
}
<Encrypted Data> //The encrypted payload that matches the JSON response normally
 returned by the TMS API, except with an unmasked payment details
```

Header Configuration

You must pass this request header to retrieve unmasked payment details: `Accept: application/jose`.

The term `application/jose` refers to Javascript Object Signing and Encryption (JOSE). JOSE is a framework that provides end-to-end security to JavaScript Object Notation (JSON)-based data structures. JOSE achieves this by offering a collection of specifications to encrypt and digitally sign JSON payloads. In this case, the response is message-level encrypted using a JSON Web Token (JWT).

Endpoint

Test: `GET https://apitest.cybersource.com/tms/v2/customers/{customerTokenId}/payment-instruments/{paymentInstrumentTokenId}`

Production: `GET https://api.cybersource.com/tms/v2/customers/{customerTokenId}/payment-instruments/{paymentInstrumentTokenId}`

Production in India: `GET https://api.in.cybersource.com/tms/v2/customers/{customerTokenId}/payment-instruments/{paymentInstrumentTokenId}`

The *{customerTokenId}* is the customer token ID returned in the **id** field when you created the customer token. The *{paymentInstrumentTokenId}* is the payment instrument token ID you want to retrieve. For more information, see Create a Customer (on page 92) and Create a Customer Payment Instrument (on page 150).

REST Example: Retrieving a Customer Payment Instrument with an Unmasked Card Number

Request

```
GET https://apitest.cybersource.com/tms/v2/customers/F2F3ADA770102B51E053A2598D0A9
078/payment-instruments/F39732BE4BDA9A1EE053AF598E0A4081
```

Response to a Successful Request


```
eyJraWQiOiJiYTE1ZDRmMTIzMTM0NjlkZjg5MDM1Nzk2YWE4Nzc4ZGM0NTY4ODlkIiwiY3R5IjoianNvbi
IsInR5cCI6IkpXVCIsImVuYyI6IkEyNTZHQ00iLCJhbGciOiJSU0EtT0FFUC0yNTYifQ.FZbQse7mPcf25
5vpZFXM4Zy8DGalqehCYUi6h6ett2WqfP2XA0uzPeRxE-P6O8Ju1trkSOJcZ4PqBcX4xwYmSs8PUmhkakn
cpjXSvYUaq4RY39kj9BRzvn47F18OWCW4CDzaTkOxi7ynGN6vb_Y3_wn5KLXAQVTUCM7Lke45oAFrCnVAC
BJtQgONKM7GZwLwRiWp_HP6D4IMUZe5Qw8Qz438scq9DQtOBum_JK_xx_IKA-r1XOkWdvnCasQnK1eEc2j
Po3EL9GDe6w3zQFEbhNtC4Rsa33-lc0lxjfuAsI8YmmtHYKeITvQ-6mB7cOP-guKnRAJk2SUPkMOL6UIfg
.zvgReLHE0ybDfWRp.bCMwJgHEV9B9xHrL9Ej1en8xZCiYEuCN6H4mcKOqxJAxZocO1AtR1xgyrfDIINAi
6Jq9UJedIvLJFyMfXx2D2x4njHmxOKzC2KJS_KTpXR1s1-pJNG68jwZ-g_zPqj1PLa_EGSu73NWhJYalGv
hDuo6Ek8bkGVUtNm9OZ89oX2KbxuWc2LQ0JbBBa_dfQWAjkRcM-URlEbhf1nvlzLwTXhRN2wfB7L1BcAsa
o51DyYXowOJpWwSK1StC2EKDXSzgpfXP1ZwSKA8SSkpVSmwOkb0n6DZNkwtKlg4eGDok9atJpbZ3qCEqqK
DYCy1levJQ7w-In2OPLpSpFGyPRUGPBMTnzQo-GtGEM1tiKbpDUzaCL_0iFSGjJCPeottP-B98R2YKdmGa
3IwyVWgzhAMJBkAfEGAx0TCWwqZE5xFW9uT2MzdX3_Mz9qBgCRa101km9dHYwajClVb1VETlHjS7zpQ1Ot
XPKmluAGTvGSr6PWn9ZiqkOd4R5LC7oA4OdVlpPhY2mJhektLOZj1uUIr5AaHyjHx-BnnFio0CDjM03t8g
l9gIeQ44ugUMwYc19Mvkiikxsvl8h9Ua2hCSFbvvq1cCOcZwb2UtI8EZcJdltw2utoiO5IbsSkE9hU2b16
QMXoVIMxiFN0OdTfJqMzJfPnyVBIkN3nmHmwLwKSek5HqdujU1hFhMJxDRdtmLD__5L0iAxuz1Sm3yx5Hm
jXWjCpzIfT9vT-pSfIdIwBakF9pBRXDSCZsAEqlwddS1DbjfNk43E_wKTmwQW96OlnUX7SK70gICydciHs
SsrElcp6lGFpbPMGs8QN1czKPrH0lrnkD21xkxhXjmC7Iqa0-XFXIU8qSV7PsBtUjOnz7oKOzXvIli2SV2
gzPEKOQ449HKPXDoBynaT5pWi4WC3JmOwAhyx2f05ABZF9-Nj_EGLe7H5EoBaCohbKkc3j24nNQ4r_n5cC
5weBCxIdkrSKh54pFQdRr72pqEW2XoOTy1Jafi3EJdC_GF0BKI3AFVw3fGEJq_rpe8PxgkkliAuywVJ43i
G_uzD-6Ib5jIA8RcDFah2jh_3tYeWws2EW3qnCuAUXREKebdGlH2BTgcgxzDn9Y6AJi3Zrdc948qxXpowi
YWr5t_5xN8x5kJcOzKVNOCzi5LggcIN-FmZsyB4rRjv9aGPrscoC1pL7xlLEnyHRnIOUy96NTG7qOQbhV3
dzawvzZN_UZ6LTyTMV9X0679NNGS2RrjxFsrYuMHdQr3SeVcTKe5FL3QBiKFgFjnYMdh73ztYW5tn6rAx2
Daq5G-FkQnD8PnHnzCplGRXopja00xEkL9lugeKxSEorDPaO8ov499M191BrTqc6XaBl7kYuelWfAoVEfC
T9FvNf28H0xA5vXJNqKFye2ExkMyk3jjfCn3pkoFwmbyha1gmaLgz788GxMyKtH9K6KMKfgSCfj-w5eJbT
l7QJeyYjFuVUqixZI024YAUoo4OrcCZag1IzLNkpOo_xOqf1iMbREnDcp2MKxMdkJWI72uB5XWztHaQPnz
BAxJcBw0_gB5AHy_AIk.ogA-QQ53MEu1VwH6_H-DQA
```


List Payment Instruments for a Customer

This section shows you how to retrieve a customer's payment instruments.

Endpoint

Test: `GET https://apitest.cybersource.com/tms/v2/customers/{customerTokenId}/payment-instruments`

Production: `GET https://api.cybersource.com/tms/v2/customers/{customerTokenId}/payment-instruments`

Production in India: `GET https://api.in.cybersource.com/tms/v2/customers/{customerTokenId}/payment-instruments`

The `{customerTokenId}` is the customer token ID returned in the **id** field when you created the customer token. For more information, see Create a Customer (on page 92) and Create a Customer Payment Instrument (on page 150).

Use these query string parameters to filter the list of payment instrument tokens:

- `offset` — Page offset number.
- `limit` — Maximum number of items you would like returned.

Required Fields for Listing Payment Instruments for a Customer

customerTokenId

Include the ID of the customer token you want to retrieve in the URL path.

Related Information

- API Field Reference for the REST API

REST Example: Listing Payment Instruments for a Customer

Request

```
GET https://apitest.cybersource.com/tms/v2/customers/7A906EC3D0F73581E0539599D30AA
DC1/payment-instruments?offset=4&limit=2
```

Response to a Successful Request


```
{
  "_links": {
    "self": {
      "href":
 "/tms/v2/customers/7A906EC3D0F73581E0539599D30AADC1/payment-instruments?offset=4&
limit=2"
    },
    "first": {
      "href":
 "/tms/v2/customers/7A906EC3D0F73581E0539599D30AADC1/payment-instruments?offset=0&
limit=2"
    },
    "prev": {
      "href":
 "/tms/v2/customers/7A906EC3D0F73581E0539599D30AADC1/payment-instruments?offset=2&
limit=2"
    },
    "next": {
      "href":
 "/tms/v2/customers/7A906EC3D0F73581E0539599D30AADC1/payment-instruments?offset=6&
limit=2"
    },
      "last": {
        "href":
 "/tms/v2/customers/7A906EC3D0F73581E0539599D30AADC1/payment-instruments?offset=8&
limit=2"
      }
  },
  "offset": 4,
  "limit": 2,
  "count": 2,
  "total": 10,
  "_embedded": {
    "paymentInstruments": [
      {
        "_links": {
          "self": {
            "href":
 "/tms/v2/customers/7A906EC3D0F73581E0539599D30AADC1/payment-instruments/7A906EC3D
0F73581E0539599D30AAPI1"
          },
          "customer": {
            "href": "/tms/v2/customers/7A906EC3D0F73581E0539599D30AADC1"
          }
        },
        "id": "7A906EC3D0F73581E0539599D30AAPI1",
        "state": "ACTIVE",
```

```
        "card": {
          "expirationMonth": "09",
          "expirationYear": "2017",
          "type": "001",
          "issueNumber": "01",
          "startMonth": "01",
          "startYear": "2016",
          "useAs": "pinless debit",
          "tokenizedInformation": {
            "requestorID": "12345",
            "transactionType": "1"
          }
        },
        "buyerInformation": {
          "companyTaxID": "12345",
          "currency": "USD",
          "dateOfBirth": "2000-12-13",
          "personalIdentification": [
            {
              "id": "57684432111321",
              "type": "driver license",
              "issuedBy": {
                "administrativeArea": "CA"
              }
            }
          ]
        },
        "billTo": {
          "firstName": "John",
          "lastName": "Smith",
          "company": "Company Name",
          "address1": "8310 Capital of Texas Highwas North",
          "address2": "Bluffstone Drive",
          "locality": "Austin",
          "administrativeArea": "TX",
          "postalCode": "78731",
          "country": "US",
          "email": "john.smith@test.com",
          "phoneNumber": "+44 2890447951"
        },
        "processingInformation": {
          "billPaymentProgramEnabled": true,
          "bankTransferOptions": {
            "SECCode": "WEB"
          }
        },
        "merchantInformation": {
          "merchantDescriptor": {
            "alternateName": "Branch Name"
```

```
          }
        },
        "metadata": {
          "creator": "mid"
        },
        "instrumentIdentifier": {
          "id": "7040000000057621111"
        },
        "_embedded": {
          "instrumentIdentifier": {
            "_links": {
              "self": {
                "href": "/tms/v1/instrumentidentifiers/7040000000057621111"
              },
              "paymentInstruments": {
                "href":
 "/tms/v1/instrumentidentifiers/7040000000057621111/paymentinstruments"
              }
            },
            "id": "7040000000057621111",
            "object": "instrumentIdentifier",
            "state": "ACTIVE",
            "card": {
              "number": "411111XXXXX1111"
            },
            "issuer": {
              "paymentAccountReference": "V0000000000004111111111111111"
            },
            "metadata": {
              "creator": "mid"
            }
          }
        }
      },
      {
        "_links": {
          "self": {
            "href":
 "/tms/v2/customers/7A906EC3D0F73581E0539599D30AAPI1/payment-instruments/7A906EC3D
0F73581E0539599D30AAPI2"
          },
          "customer": {
            "href": "/tms/v2/customers/7A906EC3D0F73581E0539599D30AAPI2"
          }
        },
        "id": "7A906EC3D0F73581E0539599D30AAPI2",
        "state": "ACTIVE",
        "card": {
          "expirationMonth": "09",
```

```
      "expirationYear": "2017",
      "type": "001",
      "issueNumber": "01",
      "startMonth": "01",
      "startYear": "2016",
      "useAs": "pinless debit",
      "tokenizedInformation": {
        "requestorID": "12345",
        "transactionType": "1"
      }
    },
    "buyerInformation": {
      "companyTaxID": "12345",
      "currency": "USD",
      "dateOfBirth": "2000-12-13",
      "personalIdentification": [
        {
          "id": "57684432111321",
          "type": "driver license",
          "issuedBy": {
            "administrativeArea": "CA"
          }
        }
      ]
    },
    "billTo": {
      "firstName": "John",
      "lastName": "Smith",
      "company": "Company Name",
      "address1": "8310 Capital of Texas Highway North",
      "address2": "Bluffstone Drive",
      "locality": "Austin",
      "administrativeArea": "TX",
      "postalCode": "78731",
      "country": "US",
      "email": "john.smith@test.com",
      "phoneNumber": "+44 2890447951"
    },
    "processingInformation": {
      "billPaymentProgramEnabled": true,
      "bankTransferOptions": {
        "SECCode": "WEB"
      }
    },
    "merchantInformation": {
      "merchantDescriptor": {
        "alternateName": "Branch Name"
      }
    },
```

```
        "metadata": {
          "creator": "mid"
        },
        "instrumentIdentifier": {
          "id": "7040000000057621111"
        },
        "_embedded": {
          "instrumentIdentifier": {
            "_links": {
              "self": {
                "href": "/tms/v1/instrumentidentifiers/7040000000057621111"
              },
              "paymentInstruments": {
                "href":
 "/tms/v1/instrumentidentifiers/7040000000057621111/paymentinstruments"
              }
            },
            "id": "7040000000057621111",
            "object": "instrumentIdentifier",
            "state": "ACTIVE",
            "card": {
              "number": "411111XXXXX1111"
            },
            "issuer": {
              "paymentAccountReference": "V0000000000004111111111111111"
            },
            "metadata": {
              "creator": "mid"
            }
          }
        }
    ]
  }
}
```


Update a Customer Payment Instrument

This section shows you how to update a customer payment instrument token.

Endpoint

Test: `PATCH https://apitest.cybersource.com/tms/v2/customers/{customerTokenId}/payment-instruments/{paymentInstrumentTokenId}`

Production: `PATCH https://api.cybersource.com/tms/v2/customers/{customerTokenId}/payment-instruments/{paymentInstrumentTokenId}`

Production in India: `PATCH https://api.in.cybersource.com/tms/v2/customers/{customerTokenId}/payment-instruments/{paymentInstrumentTokenId}`

The *{customerTokenId}* is the customer token ID returned in the **id** field when you created the customer token. The *{paymentInstrumentTokenId}* is the payment instrument token ID you want to retrieve. For more information, see Create a Customer (on page 92) and Create a Customer Payment Instrument (on page 150).

Required Fields for Updating a Customer Payment Instrument

customerTokenId

Include the ID of the customer token you want to retrieve in the URL path.

paymentInstrumentTokenId

Include the ID of the payment instrument token you want to retrieve in the URL path.

Related Information

- API Field Reference for the REST API

Optional Fields for Updating a Customer Payment Instrument

bankAccount.type

billTo.address1

billTo.address2

billTo.aminstrativeArea

billTo.company

billTo.country

billTo.email

billTo.firstName

billTo.lastName

billTo.locality

billTo.phoneNumber

billTo.postalCode

buyerInformation.companyTaxID

buyerInformation.currency

buyerInformation.dateOfBirth

buyerInformation.personalIdentification.id

buyerInformation.personalIdentification.issuedBy.administrativeArea

buyerInformation.personalIdentification.type

card.expirationMonth

card.expirationYear

card.issueNumber

card.startMonth

card.startYear

card.type

card.useAs

card.tokenizedInformation.requestorID

card.tokenizedInformation.transactionType

default

Set value to `true` if default, otherwise set value to `false`.

instrumentIdentifier.id

processingInformation.billPaymentProgramEnabled

merchantInformation.merchantDescriptor.alternateName

Related Information

- API Field Reference for the REST API

REST Example: Updating a Customer Payment Instrument

Request

```
{
  "default": "true",
  "card": {
    "expirationMonth": "12",
    "expirationYear": "2031",
    "type": "001"
  },
  "billTo": {
    "firstName": "John",
    "lastName": "Doe",
    "company": "Company Name",
    "address1": "1 Market St",
    "address2": "Unit B",
    "locality": "San Francisco",
    "administrativeArea": "CA",
    "postalCode": "94105",
    "country": "US",
    "email": "test@cybs.com",
    "phoneNumber": "4158880000"
  },
  "instrumentIdentifier": {
    "id": "7010000000016241111"
  }
}
```

Response to a Successful Request


```
{
  "_links": {
    "self": {
      "href":
 "/tms/v2/customers/F2F3ADA770102B51E053A2598D0A9078/payment-instruments/F39732BE4
BDA9A1EE053AF598E0A4081"
    },
    "customer": {
      "href": "/tms/v2/customers/F2F3ADA770102B51E053A2598D0A9078"
    }
  },
  "id": "F39732BE4BDA9A1EE053AF598E0A4081",
  "default": true,
  "state": "ACTIVE",
  "card": {
    "expirationMonth": "12",
    "expirationYear": "2031",
    "type": "001"
  },
  "billTo": {
    "firstName": "John",
    "lastName": "Doe",
    "company": "Company Name",
    "address1": "1 Market St",
    "address2": "Unit B",
    "locality": "San Francisco",
    "administrativeArea": "CA",
    "postalCode": "94105",
    "country": "US",
    "email": "test@cybs.com",
    "phoneNumber": "4158880000"
  },
  "instrumentIdentifier": {
    "id": "7010000000016241111"
  },
  "metadata": {
    "creator": "testrest"
  },
  "_embedded": {
    "instrumentIdentifier": {
      "_links": {
        "self": {
          "href": "/tms/v1/instrumentidentifiers/7010000000016241111"
        },
        "paymentInstruments": {
          "href":
 "/tms/v1/instrumentidentifiers/7010000000016241111/paymentinstruments"
```

```
        }
      },
      "id": "7010000000016241111",
      "object": "instrumentIdentifier",
      "state": "ACTIVE",
      "card": {
        "number": "411111XXXXXX1111"
      },
      "processingInformation": {
        "authorizationOptions": {
          "initiator": {
            "merchantInitiatedTransaction": {
              "previousTransactionId": "123456789012345"
            }
          }
        }
      },
      "metadata": {
        "creator": "testrest"
      }
    }
  }
}
```

Delete a Customer Payment Instrument

This section shows you how to delete a customer payment instrument token.

Endpoint

Test: `DELETE https://apitest.cybersource.com/tms/v2/customers/{customerTokenId}/payment-instruments/{paymentInstrumentTokenId}`

Production: `DELETE https://api.cybersource.com/tms/v2/customers/{customerTokenId}/payment-instruments/{paymentInstrumentTokenId}`

Production in India: `DELETE https://api.in.cybersource.com/tms/v2/customers/{customerTokenId}/payment-instruments/{paymentInstrumentTokenId}`

The *{customerTokenId}* is the customer token ID returned in the **id** field when you created the customer token. The *{paymentInstrumentTokenId}* is the payment instrument token ID you want to retrieve. For more information, see Create a Customer (on page 92) and Create a Customer Payment Instrument (on page 150).

Important: If you have more than one payment Instrument, then the default payment Instrument cannot be deleted without first selecting a new default payment instrument.

REST Example: Deleting a Customer Payment Instrument

Request

```
DELETE https://apitest.cybersource.com/tms/v2/customers/F2F3ADA770102B51E053A2598D0A9078/payment-instruments/F39732BE4BDA9A1EE053AF598E0A4081
```

Response to a Successful Request

A successful delete response returns an empty `HTTP 204 No Content` status. For more information, see HTTP Status Codes (on page 358).

Payments with Customer Payment Instrument Tokens

This section contains information on making payments with customer payment instrument tokens.

Customer payment instruments are payment instruments that are linked to a specific customer token. The following payment instruments are supported:

- Payment card
- Tokenized card (Apple Pay and Android Pay)
- ACH bank account

To process a payment using a payment instrument token, you must include the customer token ID as the value in the `paymentInformation.paymentInstrument.id` field. You can make payments using non-default payment instruments associated with the customer. For example:

- Authorizing a Payment with a Non-Default Payment Instrument (on page 198)
- Making a Credit with a Non-Default Payment Instrument (on page 202)

For more information on customer tokens and payment instrument tokens, see Customer Tokens (on page 22) and Payment Instrument Tokens (on page 22), respectively.

Authorizing a Payment with a Non-Default Payment Instrument

This section provides the information you need in order to authorize a payment with a non-default payment instrument.

Endpoint

Production: `POST https://api.cybersource.com/pts/v2/payments`

Test: `POST https://apitest.cybersource.com/pts/v2/payments`

Production in India: `POST https://api.in.cybersource.com/pts/v2/payments`

Required Fields for Authorizing a Payment with a Non-Default Payment Instrument

clientReferenceInformation.code

orderInformation.amountDetails.currency

orderInformation.amountDetails.totalAmount

paymentInformation.paymentInstrument.id

Set to the ID of the payment instrument token you want to use.

Related Information

- API field reference guide for the REST API

Optional Fields for Authorizing a Payment with a Non-Default Payment Instrument

You can use these optional fields to include additional information when authorizing a payment with a non-default payment instrument.

orderInformation.amountDetails.currency

orderInformation.amountDetails.totalAmount

orderInformation.billTo.address1

orderInformation.billTo.administrativeArea

orderInformation.billTo.country

orderInformation.billTo.email

orderInformation.billTo.firstName

orderInformation.billTo.lastName

orderInformation.billTo.locality

orderInformation.billTo.postalCode

paymentInformation.card.expirationMonth

paymentInformation.card.expirationYear

paymentInformation.card.number

paymentInformation.card.type

Related Information

- API field reference guide for the REST API

REST Example: Authorizing a Payment with a Non-Default Payment Instrument

Request

```
{
  "clientReferenceInformation": {
    "code": "12345678"
  },
    "paymentInformation": {
        "paymentInstrument": {
            "id": "0F3BB131F8143A58E063A2598D0AB921"
        }
    },
    "orderInformation": {
        "amountDetails": {
            "currency": "USD",
            "totalAmount": "10.00"
        }
    }
}
```

Response to a Successful Request


```
{
  "_links": {
    "authReversal": {
      "method": "POST",
      "href": "/pts/v2/payments/705595264858665330495 1/reversals"
    },
    "self": {
      "method": "GET",
      "href": "/pts/v2/payments/705595264858665330495 1"
    },
    "capture": {
      "method": "POST",
      "href": "/pts/v2/payments/705595264858665330495 1/captures"
    }
  },
  "clientReferenceInformation": {
    "code": "12345678"
  },
  "id": "7055952648586653304951",
  "orderInformation": {
    "amountDetails": {
      "authorizedAmount": "10.00",
      "currency": "USD"
    }
  },
  "paymentAccountInformation": {
    "card": {
      "type": "001"
    }
  },
  "paymentInformation": {
    "tokenizedCard": {
      "type": "001"
    },
    "instrumentIdentifier": {
      "id": "7010000000016241111",
      "state": "ACTIVE"
    },
    "paymentInstrument": {
      "id": "0F3BB131F8143A58E063A2598D0AB921"
    },
    "card": {
      "type": "001"
    }
  },
  "pointOfSaleInformation": {
    "terminalId": "111111"
```

```
  },
  "processorInformation": {
    "approvalCode": "888888",
    "networkTransactionId": "123456789619999",
    "transactionId": "123456789619999",
    "responseCode": "100",
    "avs": {
      "code": "X",
      "codeRaw": "I1"
    }
  },
  "reconciliationId": "67468244CRIL0U0Y",
  "status": "AUTHORIZED",
  "submitTimeUtc": "2024-01-18T16:27:45Z"
}
```

Making a Credit with a Non-Default Payment Instrument

This section shows you how to make a credit with a non-default payment instrument.

Endpoint

Test: `POST https://apitest.cybersource.com/pts/v2/credits`

Production: `POST https://api.cybersource.compts/v2/credits`

Production in India: `POST https://api.in.cybersource.compts/v2/credits`

Required Fields for Making a Credit with a Non-Default Payment Instrument

clientReferenceInformation.code

orderInformation.amountDetails.currency

orderInformation.amountDetails.totalAmount

paymentInformation.paymentInstrument.id

Set to the ID of the payment instrument token that you want to use.

Related Information

- API field reference guide for the REST API

Optional Fields for Making a Credit with a Non-Default Payment Instrument

You can use these optional fields to include additional information when making a credit with a non-default payment instrument.

orderInformation.amountDetails.currency

orderInformation.amountDetails.totalAmount

orderInformation.billTo.address1

orderInformation.billTo.administrativeArea

orderInformation.billTo.country

orderInformation.billTo.email

orderInformation.billTo.firstName

orderInformation.billTo.lastName

orderInformation.billTo.locality

orderInformation.billTo.postalCode

paymentInformation.card.expirationMonth

paymentInformation.card.expirationYear

paymentInformation.card.number

paymentInformation.card.type

Related Information

- API field reference guide for the REST API

REST Example: Making a Credit with a Non-Default Payment Instrument

Request

```
{
  "clientReferenceInformation": {
    "code": "12345678"
  },
    "paymentInformation": {
        "paymentInstrument": {
            "id": "0F3BB131F8143A58E063A2598D0AB921"
        }
    },
    "orderInformation": {
        "amountDetails": {
            "currency": "USD",
            "totalAmount": "10.00"
        }
    }
}
```

Response to a Successful Request


```
{
  "_links": {
    "void": {
      "method": "POST",
      "href": "/pts/v2/credits/705596858138644610495 3/voids"
    },
    "self": {
      "method": "GET",
      "href": "/pts/v2/credits/70559685813864461049 53"
    }
  },
  "clientReferenceInformation": {
    "code": "12345678"
  },
  "creditAmountDetails": {
    "currency": "USD",
    "creditAmount": "10.00"
  },
  "id": "705596858138644610495 3",
  "orderInformation": {
    "amountDetails": {
      "currency": "USD"
    }
  },
  "paymentAccountInformation": {
    "card": {
      "type": "001"
    }
  },
  "paymentInformation": {
    "tokenizedCard": {
      "type": "001"
    },
    "instrumentIdentifier": {
      "id": "7010000000016241111",
      "state": "ACTIVE"
    },
    "paymentInstrument": {
      "id": "0F3BB131F8143A58E063A2598D0AB921"
    },
    "card": {
      "type": "001"
    }
  },
  "processorInformation": {
    "approvalCode": "888888",
    "responseCode": "100"
```

```
  },
  "reconciliationId": "67445196PRILCQCN",
  "status": "PENDING",
  "submitTimeUtc": "2024-01-18T16:54:18Z"
}
```

Payment Instrument Tokens

The payment instrument token contains the complete billing details for the payment type including cardholder name, expiration date, and billing address. These are standalone payment instruments that cannot be associated with a customer.

Related information

Manage Payment Instrument Tokens *(on page 206)*
Payments with Payment Instrument Tokens *(on page 233)*

Manage Payment Instrument Tokens

This section contains information on managing payment instrument tokens.

A payment instrument represents a means of payment and a payment instrument token stores this information using an instrument identifier token. It does not store the card number and cannot exist without an associated instrument identifier. It can include an instrument identifier, expiration date, billing address, and card type.

You can create, retrieve, update, or delete an instrument identifier by submitting an HTTP POST, `GET`, `PATCH`, or `DELETE` operation to the `tms/v1/paymentinstruments` endpoint. Use the TMS REST API payment instrument endpoint to:

- Create a payment instrument token (on page 206)
- Retrieve a payment instrument token (on page 211)
- Update a payment instrument (on page 227)
- Delete a payment instrument (on page 231)

For more information on payment instrument tokens, see Payment Instrument Tokens (on page 22).

Create a Payment Instrument

This section shows you how to create a payment instrument token.

Endpoint

Test: `POST https://apitest.cybersource.com/tms/v1/paymentinstruments`

Production: `POST https://api.cybersource.com/tms/v1/paymentinstruments`

Production in India: `POST https://api.in.cybersource.com/tms/v1/paymentinstruments`

Required Fields for Creating a Payment Instrument

card.type

instrumentIdentifier.id

Include the ID of the instrument identifier token you want to use to create a payment instrument.

Related Information

- API Field Reference for the REST API

Optional Fields for Creating a Payment Instrument

default

Set value to `true` if default, otherwise set value to `false`.

bankAccount.type

billTo.firstName

billTo.lastName

billTo.company

billTo.address1

billTo.locality

billTo.administrativeArea

billTo.postalCode

billTo.country

billTo.email

billTo.phoneNumber

card.expirationMonth

card.expirationYear

card.issueNumber

card.startMonth

card.startYear

card.useAs

tokenizedInformation.requestorID

tokenizedInformation.transactionType

buyerInformation.companyTaxID

buyerInformation.currency

buyerInformation.dateOfBirth

buyerInformation.personalIdentification.id

buyerInformation.personalIdentification.type

buyerInformation.personalIdentification.issuedBy.administrativeArea

billTo.address2

processingInformation.billPaymentProgramEnabled

processingInformation.bankTransferOptions.SECCode

merchantInformation.merchantDescriptor.alternateName

Related Information

- API Field Reference for the REST API

REST Example: Creating a Payment Instrument

Request

```
{
  "card": {
    "expirationMonth": "12",
    "expirationYear": "2031",
    "type": "visa"
  },
  "billTo": {
    "firstName": "John",
    "lastName": "Doe",
    "company": "Company Name",
    "address1": "1 Market St",
    "locality": "San Francisco",
    "administrativeArea": "CA",
    "postalCode": "94105",
    "country": "US",
    "email": "test@cybs.com",
    "phoneNumber": "4158880000"
  },
  "instrumentIdentifier": {
    "id": "7010000000016241111"
  }
}
```

Response to a Successful Request

```
{
  "_links": {
    "self": {

 "href": https://apitest.cybersource.com/tms/v1/paymentinstruments/F39763E8CFDF235
4E053AF598E0AF684"
    }
  },
  "id": "F39763E8CFDF2354E053AF598E0AF684",
  "object": "paymentInstrument",
  "state": "ACTIVE",
  "card": {
    "expirationMonth": "12",
    "expirationYear": "2031",
    "type": "visa"
  },
  "billTo": {
    "firstName": "John",
    "lastName": "Doe",
    "company": "Company Name",
    "address1": "1 Market St",
    "locality": "San Francisco",
    "administrativeArea": "CA",
    "postalCode": "94105",
    "country": "US",
    "email": "test@cybs.com",
    "phoneNumber": "4158880000"
  },
  "metadata": {
    "creator": "testrest"
  },
  "_embedded": {
    "instrumentIdentifier": {
      "_links": {
        "self": {
          "href":
 "https://apitest.cybersource.com/tms/v1/instrumentidentifiers/70100000000162411
11"
        },
        "paymentInstruments": {
          "href":
 "https://apitest.cybersource.com/tms/v1/instrumentidentifiers/70100000000162411
11/paymentinstruments"
        }
      },
      "id": "7010000000016241111",
      "object": "instrumentIdentifier",
```

```
      "state": "ACTIVE",
      "card": {
        "number": "411111XXXXXX1111"
      },
      "processingInformation": {
        "authorizationOptions": {
          "initiator": {
            "merchantInitiatedTransaction": {
              "previousTransactionId": "123456789619999"
            }
          }
        }
      },
      "metadata": {
        "creator": "testrest"
      }
    }
  }
}
```

Retrieve a Payment Instrument

This section shows you how to retrieve a payment instrument token.

Endpoint

Test: `GET https://apitest.cybersource.com/tms/v1/paymentinstruments/{paymentInstrumentTokenId}`

Production: `GET https://api.cybersource.com/tms/v1/paymentinstruments/{paymentInstrumentTokenId}`

Production in India: `GET https://api.in.cybersource.com/tms/v1/paymentinstruments/{paymentInstrumentTokenId}`

The *{paymentInstrumentTokenId}* is the payment instrument token ID you want to retrieve. For more information, see Create a Payment Instrument (on page 206).

REST Example: Retrieving a Payment Instrument

Request

```
GET https://apitest.cybersource.com/tms/v1/paymentinstruments/F39763E8CFDF2354E053AF598E0AF684
```

Response to a Successful Request


```
{
  "_links": {
    "self": {
      "href":
 "https://apitest.cybersource.comtms/v1/paymentinstruments/F39763E8CFDF2354E053AF5
98E0AF684"
    }
  },
  "id": "F39763E8CFDF2354E053AF598E0AF684",
  "object": "paymentInstrument",
  "state": "ACTIVE",
  "card": {
    "expirationMonth": "12",
    "expirationYear": "2031",
    "type": "visa"
  },
  "billTo": {
    "firstName": "John",
    "lastName": "Doe",
    "company": "Company Name",
    "address1": "1 Market St",
    "locality": "San Francisco",
    "administrativeArea": "CA",
    "postalCode": "94105",
    "country": "US",
    "email": "test@cybs.com",
    "phoneNumber": "4158880000"
  },
  "metadata": {
    "creator": "testrest"
  },
  "_embedded": {
    "instrumentIdentifier": {
      "_links": {
        "self": {
          "href":
 "https://apitest.cybersource.com/tms/v1/instrumentidentifiers/70100000000162411
11"
        },
        "paymentInstruments": {
          "href":
 "https://apitest.cybersource.comtms/v1/instrumentidentifiers/70100000000162411
11/paymentinstruments"
        }
      },
      "id": "7010000000016241111",
      "object": "instrumentIdentifier",
```

```
      "state": "ACTIVE",
      "card": {
        "number": "411111XXXXXX1111"
      },
      "processingInformation": {
        "authorizationOptions": {
          "initiator": {
            "merchantInitiatedTransaction": {
              "previousTransactionId": "123456789619999"
            }
          }
        }
      },
      "metadata": {
        "creator": "testrest"
      }
    }
  }
}
```

Find Payment Instruments by Card Number

This section shows you how to find payment instruments by card number.

Endpoint

Test: `GET https://apitest.cybersource.com/tms/v1/instrumentidentifiers/{instrumentIdentifierTokenId}/paymentinstruments?offset=0&limit=20`

Production: `GET https://api.cybersource.com/tms/v1/instrumentidentifiers/{instrumentIdentifierTokenId}/paymentinstruments?offset=0&limit=20`

Production in India: `GET https://api.in.cybersource.com/tms/v1/instrumentidentifiers/{instrumentIdentifierTokenId}/paymentinstruments?offset=0&limit=20`

`instrumentIdentifierTokenId` is the instrument identifier token ID returned in the **id** field when you created the instrument identifier token. For more information, see Create an Instrument Identifier (on page 243).

Use these query string parameters to filter the list of payment instrument tokens:

- `offset` — Page offset number.
- `limit` — Maximum number of items you would like returned.

Required Fields for Finding Payment Instruments by Card Number

instrumentIdentifierTokenId

Include the ID of the instrument identifier token you want to retrieve in the URL path.

Related Information

- API Field Reference for the REST API

REST Example: Finding Payment Instruments by Card Number

Request

```
GET https://apitest.cybersource.com/tms/v1/instrumentidentifiers/7010000000016241111/paymentinstruments?offset=0&limit=5
```

Response to a Successful Request


```
{
    "_links": {
        "self": {
            "href":
 "https://apitest.cybersource.com/tms/v1/instrumentidentifiers/701000000001624111
11/paymentinstruments?offset=0&limit=5"
        },
        "first": {
            "href":
 "https://apitest.cybersource.comtms/v1/instrumentidentifiers/701000000001624111
11/paymentinstruments?offset=0&limit=5"
        },
        "next": {
            "href":
 "https://apitest.cybersource.com/tms/v1/instrumentidentifiers/701000000001624111
11/paymentinstruments?offset=5&limit=5"
        },
        "last": {
            "href":
 "https://apitest.cybersource.com/tms/v1/instrumentidentifiers/701000000001624111
11/paymentinstruments?offset=121265&limit=5"
        }
    },
    "object": "collection",
    "offset": 0,
    "limit": 5,
    "count": 5,
    "total": 121269,
    "_embedded": {
        "paymentInstruments": [
            {
                "_links": {
                    "self": {
                        "href":
 "https://apitest.cybersource.com/tms/v1/paymentinstruments/F4D5E715F7BD9910E053A2
598D0A7278"
                    },
                    "customer": {
                        "href":
 "https://apitest.cybersource.com/tms/v1/customers/F4D5E715F75E9910E053A2598D0A72
78"
                    }
                },
                "id": "F4D5E715F7BD9910E053A2598D0A7278",
                "object": "paymentInstrument",
                "state": "ACTIVE",
                "card": {
```

```
                    "expirationMonth": "12",
                    "expirationYear": "2031",
                    "type": "visa"
                },
                "billTo": {
                    "firstName": "John",
                    "lastName": "Doe",
                    "company": "Visa",
                    "address1": "1 Market St",
                    "locality": "san francisco",
                    "administrativeArea": "CA",
                    "postalCode": "94105",
                    "country": "US",
                    "email": "test@cybs.com",
                    "phoneNumber": "4158880000"
                },
                "metadata": {
                    "creator": "testrest"
                },
                "_embedded": {
                    "instrumentIdentifier": {
                        "_links": {
                            "self": {
                                "href":
 "https://apitest.cybersource.com/tms/v1/instrumentidentifiers/70100000000162411
11"
                            },
                            "paymentInstruments": {
                                "href":
 "https://apitest.cybersource.com/tms/v1/instrumentidentifiers/70100000000162411
11/paymentinstruments"
                            }
                        },
                        "id": "7010000000016241111",
                        "object": "instrumentIdentifier",
                        "state": "ACTIVE",
                        "card": {
                            "number": "411111XXXXXX1111"
                        },
                        "processingInformation": {
                            "authorizationOptions": {
                                "initiator": {
                                    "merchantInitiatedTransaction": {
                                        "previousTransactionId": "123456789619999"
                                    }
                                }
                            }
                        },
                        "metadata": {
```

```
                            "creator": "testrest"
                        }
                    }
                }
            },
            {
                "_links": {
                    "self": {
                        "href":
 "https://apitest.cybersource.com/tms/v1/paymentinstruments/F4D5E70505B30CF9E053AF
598E0A005F"
                    },
                    "customer": {
                        "href":
 "https://apitest.cybersource.com/tms/v1/customers/F4D5E70505B40CF9E053AF598E0A00
5F"
                    }
                },
                "id": "F4D5E70505B30CF9E053AF598E0A005F",
                "object": "paymentInstrument",
                "state": "ACTIVE",
                "card": {
                    "expirationMonth": "02",
                    "expirationYear": "2024",
                    "type": "visa"
                },
                "buyerInformation": {
                    "currency": "USD"
                },
                "billTo": {
                    "firstName": "NOREAL",
                    "lastName": "NAME",
                    "address1": "1295 Charleston Road",
                    "locality": "Mountain View",
                    "administrativeArea": "CA",
                    "postalCode": "94043",
                    "country": "US",
                    "email": "customer_email=null@cybersource.com"
                },
                "metadata": {
                    "creator": "testrest"
                },
                "_embedded": {
                    "instrumentIdentifier": {
                        "_links": {
                            "self": {
                                "href":
 "https://apitest.cybersource.com/tms/v1/instrumentidentifiers/70100000000162411
11"
```

```
                            },
                            "paymentInstruments": {
                                "href":
 "https://apitest.cybersource.com/tms/v1/instrumentidentifiers/70100000000162411
11/paymentinstruments"
                            }
                        },
                        "id": "7010000000016241111",
                        "object": "instrumentIdentifier",
                        "state": "ACTIVE",
                        "card": {
                            "number": "411111XXXXXX1111"
                        },
                        "processingInformation": {
                            "authorizationOptions": {
                                "initiator": {
                                    "merchantInitiatedTransaction": {
                                        "previousTransactionId": "123456789619999"
                                    }
                                }
                            }
                        },
                        "metadata": {
                            "creator": "testrest"
                        }
                    }
                }
            },
            {
                "_links": {
                    "self": {
                        "href":
 "https://apitest.cybersource.com/tms/v1/paymentinstruments/F4D566EED6D369CCE053AF
598E0A495B"
                    },
                    "customer": {
                        "href":
 "https://apitest.cybersource.com/tms/v1/customers/F4D5523603862EE0E053AF598E0AE5
FE"
                    }
                },
                "id": "F4D566EED6D369CCE053AF598E0A495B",
                "object": "paymentInstrument",
                "state": "ACTIVE",
                "card": {
                    "expirationMonth": "12",
                    "expirationYear": "2031",
                    "type": "visa"
                },
```

```
                "buyerInformation": {
                    "currency": "USD"
                },
                "billTo": {
                    "firstName": "JOHN",
                    "lastName": "DOE",
                    "address1": "1 Market St",
                    "locality": "san francisco",
                    "administrativeArea": "CA",
                    "postalCode": "94105",
                    "country": "US",
                    "email": "test@cybs.com",
                    "phoneNumber": "4158880000"
                },
                "processingInformation": {
                    "billPaymentProgramEnabled": false
                },
                "metadata": {
                    "creator": "testrest"
                },
                "_embedded": {
                    "instrumentIdentifier": {
                        "_links": {
                            "self": {
                                "href":
 "https://apitest.cybersource.com/tms/v1/instrumentidentifiers/70100000000162411
11"
                            },
                            "paymentInstruments": {
                                "href":
 "https://apitest.cybersource.com/tms/v1/instrumentidentifiers/70100000000162411
11/paymentinstruments"
                            }
                        },
                        "id": "7010000000016241111",
                        "object": "instrumentIdentifier",
                        "state": "ACTIVE",
                        "card": {
                            "number": "411111XXXXXX1111"
                        },
                        "processingInformation": {
                            "authorizationOptions": {
                                "initiator": {
                                    "merchantInitiatedTransaction": {
                                        "previousTransactionId": "123456789619999"
                                    }
                                }
                            }
                        },
```

```
                        "metadata": {
                            "creator": "testrest"
                        }
                    }
                }
            },
            {
                "_links": {
                    "self": {
                        "href":
 "https://apitest.cybersource.com/tms/v1/paymentinstruments/F4CDBDD6E0A57EC9E053AF
598E0AB69F"
                    },
                    "customer": {
                        "href":
 "https://apitest.cybersource.com/tms/v1/customers/F4CDBCA630247B2EE053AF598E0ADC
91"
                    }
                },
                "id": "F4CDBDD6E0A57EC9E053AF598E0AB69F",
                "object": "paymentInstrument",
                "state": "ACTIVE",
                "card": {
                    "expirationMonth": "12",
                    "expirationYear": "2034",
                    "type": "visa"
                },
                "buyerInformation": {
                    "currency": "USD"
                },
                "billTo": {
                    "firstName": "JOHN",
                    "lastName": "DOE",
                    "address1": "1 Market St",
                    "locality": "san francisco",
                    "administrativeArea": "CA",
                    "postalCode": "94105",
                    "country": "US",
                    "email": "test@cybs.com",
                    "phoneNumber": "4158880000"
                },
                "processingInformation": {
                    "billPaymentProgramEnabled": false
                },
                "metadata": {
                    "creator": "testrest"
                },
                "_embedded": {
                    "instrumentIdentifier": {
```

```
                        "_links": {
                            "self": {
                                "href":
 "https://apitest.cybersource.com/tms/v1/instrumentidentifiers/70100000000162411
11"
                            },
                            "paymentInstruments": {
                                "href":
 "https://apitest.cybersource.com/tms/v1/instrumentidentifiers/70100000000162411
11/paymentinstruments"
                            }
                        },
                        "id": "7010000000016241111",
                        "object": "instrumentIdentifier",
                        "state": "ACTIVE",
                        "card": {
                            "number": "411111XXXXXX1111"
                        },
                        "processingInformation": {
                            "authorizationOptions": {
                                "initiator": {
                                    "merchantInitiatedTransaction": {
                                        "previousTransactionId": "123456789619999"
                                    }
                                }
                            }
                        },
                        "metadata": {
                            "creator": "testrest"
                        }
                    }
                }
            },
            {
                "_links": {
                    "self": {
                        "href":
 "https://apitest.cybersource.com/tms/v1/paymentinstruments/F4CDEF212EAA0B13E053AF
598E0AB8F4"
                    },
                    "customer": {
                        "href":
 "https://apitest.cybersource.com/tms/v1/customers/F4CDBCA630247B2EE053AF598E0ADC
91"
                    }
                },
                "id": "F4CDEF212EAA0B13E053AF598E0AB8F4",
                "object": "paymentInstrument",
                "state": "ACTIVE",
```

```
                "card": {
                    "expirationMonth": "12",
                    "expirationYear": "2031",
                    "type": "visa"
                },
                "buyerInformation": {
                    "currency": "USD"
                },
                "billTo": {
                    "firstName": "JOHN",
                    "lastName": "DOE",
                    "address1": "1 Market St",
                    "locality": "san francisco",
                    "administrativeArea": "CA",
                    "postalCode": "94105",
                    "country": "US",
                    "email": "test@cybs.com",
                    "phoneNumber": "4158880000"
                },
                "processingInformation": {
                    "billPaymentProgramEnabled": false
                },
                "metadata": {
                    "creator": "testrest"
                },
                "_embedded": {
                    "instrumentIdentifier": {
                        "_links": {
                            "self": {
                                "href":
 "https://apitest.cybersource.com/tms/v1/instrumentidentifiers/70100000000162411
11"
                            },
                            "paymentInstruments": {
                                "href":
 "https://apitest.cybersource.com/tms/v1/instrumentidentifiers/70100000000162411
11/paymentinstruments"
                            }
                        },
                        "id": "7010000000016241111",
                        "object": "instrumentIdentifier",
                        "state": "ACTIVE",
                        "card": {
                            "number": "411111XXXXXX1111"
                        },
                        "processingInformation": {
                            "authorizationOptions": {
                                "initiator": {
                                    "merchantInitiatedTransaction": {
```

```
                                        "previousTransactionId": "123456789619999"
                                    }
                                }
                            }
                        },
                        "metadata": {
                            "creator": "testrest"
                        }
                    }
                }
            }
        ]
    }
}
```

Retrieve a Payment Instrument with an Unmasked Card Number

This section shows you how to retrieve a payment instrument with an unmasked card number.

Important: To retrieve unmasked payment details, you must ensure that your MLE key pair and your token vault are configured correctly. For more information on MLE keys, see Message-Level Encryption Keys (on page 65). For more information on token vaults, see Token Vault Management (on page 62). If necessary, contact your Cybersource account manager or customer support.

The response is BASE 64-encoded JSON web encryption (JWE) token. The decoded JWE has these elements:

```
{ "alg": "RSA-OAEP-256", //The algorithm used to encrypt the CEK
    "cty": "json", //The content type
    "typ": "JWT", //The token type
    "enc": "A256GCM", //The algorithm that is used to encrypt the message
    "kid": "keyId" //The serial number of shared public cert for encryption of CEK
}
<Encrypted Data> //The encrypted payload that matches the JSON response normally
 returned by the TMS API, except with an unmasked payment details
```

Header Configuration

You must pass this request header to retrieve unmasked payment details: `Accept: application/jose`.

The term `application/jose` refers to Javascript Object Signing and Encryption (JOSE). JOSE is a framework that provides end-to-end security to JavaScript Object Notation (JSON)-based data structures. JOSE achieves this by offering a collection of specifications to encrypt and digitally sign JSON payloads. In this case, the response is message-level encrypted using a JSON Web Token (JWT).

Endpoint

Test: `GET https://apitest.cybersource.com/tms/v1/paymentinstruments/{paymentInstrumentTokenId}`

Production: `GET https://api.cybersource.com/tms/v1/paymentinstruments/{paymentInstrumentTokenId}`

Production in India: `GET https://api.in.cybersource.com/tms/v1/paymentinstruments/{paymentInstrumentTokenId}`

The *{paymentInstrumentTokenId}* is the payment instrument token ID you want to retrieve. For more information, see Create a Payment Instrument (on page 206).

REST Example: Retrieving a Payment Instrument with an Unmasked Card Number

Request

```
GET https://apitest.cybersource.com/tms/v1/paymentinstruments/F39763E8CFDF2354E053
AF598E0AF684
```

Response to a Successful Request


```
eyJraWQiOiJiYTE1ZDRmMTIzMTM0NjlkZjg5MDM1Nzk2YWE4Nzc4ZGM0NTY4ODlkIiwiY3R5IjoianNvbi
IsInR5cCI6IkpXVCIsImVuYyI6IkEyNTZHQ00iLCJhbGciOiJSU0EtT0FFUC0yNTYifQ.sPdY7WNX0jWad
y6jdHytYj8WmAyBLq401OeNNc3-cb2LWyOo11LY3r2mcJo_foZN3W175B0LzEH1IgYT96eQ8qpct7UXvvQ
LqdB25XCQIsRMU0tqugAox9QKDk1q0DMZpeki8O0_HWu1Nk6vGKwQP2XrCPDs9eZ4tHQCoId_8bffjHFOO
gVDgFtG7pJ0MpIC7eeadJZCQKsEp5ZEwzTaGmyJrVLpwMKPMKh63eLYNSTndQihOpMXlWZS1yMvZJzgdf1
8qQdBB-bINs2jGhgTAGhPaorRsFpHPqvOQm9WRAMFJU0nq1zzEd2xyf9nG45Wl6VXEOZi87c7riyketm0l
A.mxf5cgCFa2NwXuS1._NBdqbu4r0glrtKkIOLkGdiu1AzmbNLloKFY7-tMIaQ2xA1IpgR1tDHQOfQbumP
S752jtjPPvXpHnXHp3pxifM8TJR_F76NcMI0SO3r5_PLePiLtQeyZJnaW6o6ENTrFNnhgG80TNLFS4NqsX
6sIQsgm5D2S2Kf1yQL4B3goxHJTMulngvTBVJBGUf6rw5zr-q-w7buAA8NoquyIfGx0wORSFO8e_392aX5
AWbySFoUobJ1arQJARsfKdoyHsskmfsCJAwRZF6_uvFbw7uoq60TmBjwfGTdsJffqGEYSuU8ZzXWl6Q9sP
AGptwHj-JuWVnEZAq3362Cgqv3DfZtEAS-ewBd8DpzCQXatC4pUz1xj3sE5RzBQtzt8IY_exCL970UsMxh
5pJc8eqT61z5Sf7nxaNj1limcnH7_rnR1LaJ8foQAvZo8rJl8PuLe3inNOqYhAMpu6UURNB126LPHi0W7F
7o4MtFa8fm5rNF7Mbk-z4Xwx4b-FNKr3g_5JyYbJgOSAF9Kwbg7GzOGLyIPwTvXpUbFkyWoGWCgvfRDTVT
BrbdlwcuFFDlFA4B-i9L79DcMRgHb6VqgVuy-A4fA9990ctmwChY0kkfnJYcFcEaT0bgLpJw6hadtmGgyW
62yJMCRLF4GtU_PUyZ6k9s64KH6Ulro7Cbu_wcXiqklqymgCr50Ifx-gDtL3tAv_YoiI-numvNsk0EY8JZ
tmI4YOUK9V1SbrVy354X3rhPzUt2k5F8LHnExOnKugsACoFFOjpDaEBe1kI7X6UfKZU876m1H0lS5-ccZ3
VVDbdpxHlMKrSrMehno6g4ba9phdqtvuajef6P73yF3kvHoHhXvxxrEUYcUGIlef9HMvN6NS9sxj2j5Q9L
sED1XuyGBoRd5JO-gjeIHx0P__6SPiX2WdfqvVNIaS9f1k1eOOmCTERz9Esmb7vS13XzKyaN5DAFM9R3Tl
2PUmDN-AtlRN5A9moTsvvQKci-CsGTUeEYUWfpbU4UHtzXic06BygAaFJat1plfeJy1u1qhVLUWkC4Jo6i
_KryvAiJ8qb8urb_TFGWHhs2-JLQs8e3e20Ze4AkayQW-hypmshrsMgi86mqcD4o7IOY_27H4PYD2rVNPw
0lSUuxDYCp87ILro_ROiMKu3Gi1jlX-MDqJ-v3PCf21EjgEsB8kV4L1O8ZKFCVOFGpVaXTvhKXjeEUyYVS
B9uM6UTWcYGJznkClJV3vio1xpnRVpeppmTc4x0FABt7xPXCI_B81Q2q2mR9MS_Az7l-XTRIPz6skcMMjS
yS6HI4f-zg__TVWw78gupg4O0xFT1Mpbcs3HxQPtcBHoQ3EWenIcB1Pnso9IOwN3z4bSDj1OI3-cgMRFPw
UKLhJvOh0I5Jql_BKSdEnTJ4WyqY2EswrlG7dZ4fVexoMOi9UX117GqQcmdj0mboOnXDPKfclfv55nkg7o
gHhz5OvsonLxXA9LwCnL0iyISowImm1pUc-Gx1gnEPXvx1Xew7ARamkJIam3MAqhLmxwE0E6CO9xw8AG3w
DSznPK3RyE6JeiuVxhRbr5hJGQLIfH-gu13NTMh3JtPNsnmz0uvF2nZKmcWj8QmuHE76L3qYD7xCwXbGwS
DPHp7AhAPueCG8D0sG6Ilf_0S9P3-mTM1PhL2_AFpF_r9L-R-3-QrgJXVGYTbQaIFJvGG_swpS_o6s2c9i
EKI7WK3nZG0pfjiFw0UGTF4cNEj2DWzgLCj21pcKgqUDbncf3hYbqnHgNUmxHGjjOxZdiaL31-ccfNodHg
6O8kvRr_hhEA9IKG49uCJoPqJtJmOFa4MuSEdIuWBF_lSc.fenpFUQKAgR4qz7Ft_6Igg
```


Update a Payment Instrument

This section shows you how to update a payment instrument token.

Endpoint

Test: `PATCH https://apitest.cybersource.com/tms/v1/paymentinstruments/{paymentInstrumentTokenId}`

Production: `PATCH https://api.cybersource.com/tms/v1/paymentinstruments/{paymentInstrumentTokenId}`

Production in India: `PATCH https://api.in.cybersource.com/tms/v1/paymentinstruments/{paymentInstrumentTokenId}`

The *{paymentInstrumentTokenId}* is the payment instrument token ID you want to retrieve. For more information, see Create a Payment Instrument (on page 206).

Optional Fields for Updating a Payment Instrument

default

bankAccount.type

card.issueNumber

card.startMonth

card.startYear

card.useAs

tokenizedInformation.requestorID

tokenizedInformation.transactionType

buyerInformation.companyTaxID

buyerInformation.currency

buyerInformation.dateOfBirth

buyerInformation.personalIdentification.id

buyerInformation.personalIdentification.type

buyerInformation.personalIdentification.issuedBy.administrativeArea

billTo.address2

processingInformation.billPaymentProgramEnabled

processingInformation.bankTransferOptions.SECCode

merchantInformation.merchantDescriptor.alternateName

Related Information

- API Field Reference for the REST API

REST Example: Updating a Payment Instrument

Request

```
PATCH https://apitest.cybersource.com/tms/v1/paymentinstruments/F39763E8CFDF2354E053AF598E0AF684

{
  "card": {
    "expirationMonth": "12",
    "expirationYear": "2031",
    "type": "visa"
  },
  "billTo": {
    "firstName": "John",
    "lastName": "Doe",
    "company": "Company Name",
    "address1": "1 Market St",
    "address2": "Unit B"
    "locality": "San Francisco",
    "administrativeArea": "CA",
    "postalCode": "94105",
    "country": "US",
    "email": "test@cybs.com",
    "phoneNumber": "4158880000"
  },
  "instrumentIdentifier": {
    "id": "7010000000016241111"
  }
}
```

Response to a Successful Request


```
{
  "_links": {
    "self": {
      "href":
 "https://apitest.cybersource.com/tms/v1/paymentinstruments/F39763E8CFDF2354E053AF
598E0AF684"
    }
  },
  "id": "F39763E8CFDF2354E053AF598E0AF684",
  "object": "paymentInstrument",
  "state": "ACTIVE",
  "card": {
    "expirationMonth": "12",
    "expirationYear": "2031",
    "type": "visa"
  },
  "billTo": {
    "firstName": "Jack",
    "lastName": "Smith",
    "company": "Company Name",
    "address1": "1 Market St",
    "address2": "Unit B",
    "locality": "San Francisco",
    "administrativeArea": "CA",
    "postalCode": "94105",
    "country": "US",
    "email": "updatedemail@vas.com",
    "phoneNumber": "4158888674"
  },
  "metadata": {
    "creator": "testrest"
  },
  "_embedded": {
    "instrumentIdentifier": {
      "_links": {
        "self": {
          "href":
 "https://apitest.cybersource.com/tms/v1/instrumentidentifiers/70100000000162411
11"
        },
        "paymentInstruments": {
          "href":
 "https://apitest.cybersource.com/tms/v1/instrumentidentifiers/70100000000162411
11/paymentinstruments"
        }
      },
      "id": "7010000000016241111",
```

```
      "object": "instrumentIdentifier",
      "state": "ACTIVE",
      "card": {
        "number": "411111XXXXXX1111"
      },
      "processingInformation": {
        "authorizationOptions": {
          "initiator": {
            "merchantInitiatedTransaction": {
              "previousTransactionId": "123456789619999"
            }
          }
        }
      },
      "metadata": {
        "creator": "testrest"
      }
    }
  }
}
```

Delete a Payment Instrument

This section shows you how to delete a payment instrument token.

Endpoint

Test: `DELETE https://apitest.cybersource.com/tms/v1/paymentinstruments/{paymentInstrumentTokenId}`

Production: `DELETE https://api.cybersource.com/tms/v1/paymentinstruments/{paymentInstrumentTokenId}`

Production in India: `DELETE https://api.in.cybersource.com/tms/v1/paymentinstruments/{paymentInstrumentTokenId}`

The *{paymentInstrumentTokenId}* is the payment instrument token ID you want to retrieve. For more information, see Create a Payment Instrument (on page 206).

Required Fields for Deleting a Payment Instrument

paymentInstrumentTokenId

Include the ID of the payment instrument token you want to retrieve in the URL path.

Related Information

- API Field Reference for the REST API

REST Example: Deleting a Payment Instrument

Request

```
DELETE https://apitest.cybersource.com/tms/v1/paymentinstruments/F39763E8CFDF2354E053AF598E0AF684
```

Successful Response

A successful delete response returns an empty `HTTP 204 No Content` status. For more information, see HTTP Status Codes (on page 358).

Payments with Payment Instrument Tokens

This section contains information on making payments with payment instrument tokens.

A payment instrument represents a means of payment and a payment instrument token stores this information using an instrument identifier token. It does not store the card number and cannot exist without an associated instrument identifier. It can include an instrument identifier, expiration date, billing address, and card type. In the case of non-network token transactions, you can use card or bank account information fields with a payment instrument to make a payment transaction.

To process a payment using a payment instrument token, you must include the customer token ID as the value in the `paymentInformation.paymentInstrument.id` field. For example:

- Authorizing a Payment with a Payment Instrument (on page 234)
- Making a Credit with a Payment Instrument (on page 238)

For more information on payment instrument tokens, see Payment Instrument Tokens (on page 22).

Authorizing a Payment with a Payment Instrument

This section provides the information you need in order to authorize a payment with a payment instrument.

Endpoint

Production: `POST https://api.cybersource.com/pts/v2/payments`

Test: `POST https://apitest.cybersource.com/pts/v2/payments`

Production in India: `POST https://api.in.cybersource.com/pts/v2/payments`

Required Fields for Authorizing a Payment with a Payment Instrument

clientReferenceInformation.code

orderInformation.amountDetails.currency

orderInformation.amountDetails.totalAmount

paymentInformation.paymentInstrument.id

Set to the ID of the payment instrument token you want to use.

Related Information

- API field reference guide for the REST API

Optional Fields for Authorizing a Payment with a Payment Instrument

You can use these optional fields to include additional information when authorizing a payment with a payment instrument.

orderInformation.amountDetails.currency

orderInformation.amountDetails.totalAmount

orderInformation.billTo.address1

orderInformation.billTo.administrativeArea

orderInformation.billTo.country

orderInformation.billTo.email

orderInformation.billTo.firstName

orderInformation.billTo.lastName

orderInformation.billTo.locality

orderInformation.billTo.postalCode

paymentInformation.card.expirationMonth

paymentInformation.card.expirationYear

paymentInformation.card.number

paymentInformation.card.type

Related Information

- API field reference guide for the REST API

REST Example: Authorizing a Payment with a Payment Instrument

Request

```
{
  "clientReferenceInformation": {
    "code": "12345678"
  },
    "paymentInformation": {
        "paymentInstrument": {
            "id": "F4D5E715F7BD9910E053A2598D0A7278"
        }
    },
    "orderInformation": {
        "amountDetails": {
            "currency": "USD",
            "totalAmount": "10.00"
        }
    }
}
```

Response to a Successful Request


```
{
    "_links": {
        "authReversal": {
            "method": "POST",
            "href": "/pts/v2/payments/676571362873613810395 5/reversals"
        },
        "self": {
            "method": "GET",
            "href": "/pts/v2/payments/676571362873613810395 5"
        },
        "capture": {
            "method": "POST",
            "href": "/pts/v2/payments/676571362873613810395 5/captures"
        }
    },
    "clientReferenceInformation": {
        "code": "12345678"
    },
    "id": "676571362873613810395 5",
    "orderInformation": {
        "amountDetails": {
            "authorizedAmount": "10.00",
            "currency": "USD"
        }
    },
    "paymentAccountInformation": {
        "card": {
            "type": "001"
        }
    },
    "paymentInformation": {
        "tokenizedCard": {
            "type": "001"
        },
        "instrumentIdentifier": {
            "id": "7010000000016241111",
            "state": "ACTIVE"
        },
        "paymentInstrument": {
            "id": "F4D5E715F7BD9910E053A2598D0A7278"
        },
        "card": {
            "type": "001"
        },
        "customer": {
            "id": "F4D5E715F75E9910E053A2598D0A7278"
        }
```

```
    },
    "pointOfSaleInformation": {
        "terminalId": "111111"
    },
    "processorInformation": {
        "approvalCode": "888888",
        "networkTransactionId": "123456789619999",
        "transactionId": "123456789619999",
        "responseCode": "100",
        "avs": {
            "code": "X",
            "codeRaw": "I1"
        }
    },
    "reconciliationId": "60561224BE37KN5W",
    "status": "AUTHORIZED",
    "submitTimeUtc": "2023-02-16T18:16:03Z"
}
```

Making a Credit with a Payment Instrument

This section shows you how to make a credit with a payment instrument.

Endpoint

Test: `POST https://apitest.cybersource.com/pts/v2/credits`

Production: `POST https://api.cybersource.compts/v2/credits`

Production in India: `POST https://api.in.cybersource.compts/v2/credits`

Required Fields for Making a Credit with a Payment Instrument

clientReferenceInformation.code

orderInformation.amountDetails.currency

orderInformation.amountDetails.totalAmount

paymentInformation.paymentInstrument.id

Set to the ID of the payment instrument token you want to use.

Related Information

- API field reference guide for the REST API

REST Example: Making a Credit with a Payment Instrument

Request

```
{
  "clientReferenceInformation": {
    "code": "12345678"
  },
    "paymentInformation": {
        "paymentInstrument": {
            "id": "F4D5E715F7BD9910E053A2598D0A7278"
        }
    },
    "orderInformation": {
```

```
        "amountDetails": {
            "currency": "USD",
            "totalAmount": "10.00"
        }
    }
}
```

Response to a Successful Request


```
{
  "_links": {
    "void": {
      "method": "POST",
      "href": "/pts/v2/credits/705596958668646710495 3/voids"
    },
    "self": {
      "method": "GET",
      "href": "/pts/v2/credits/705596958668646710495 3"
    }
  },
  "clientReferenceInformation": {
    "code": "12345678"
  },
  "creditAmountDetails": {
    "currency": "USD",
    "creditAmount": "10.00"
  },
  "id": "7055969586686467104953",
  "orderInformation": {
    "amountDetails": {
      "currency": "USD"
    }
  },
  "paymentAccountInformation": {
    "card": {
      "type": "001"
    }
  },
  "paymentInformation": {
    "tokenizedCard": {
      "type": "001"
    },
    "instrumentIdentifier": {
      "id": "7010000000016241111",
      "state": "ACTIVE"
    },
    "paymentInstrument": {
      "id": "F4D5E715F7BD9910E053A2598D0A7278"
    },
    "card": {
      "type": "001"
    }
  },
  "processorInformation": {
    "approvalCode": "888888",
    "responseCode": "100"
```

```
  },
  "reconciliationId": "67446174JRIKXXHB",
  "status": "PENDING",
  "submitTimeUtc": "2024-01-18T16:55:59Z"
}
```

Instrument Identifier Tokens

Instrument identifier tokens represent tokenized payment account numbers. Tokenized payment account information includes a primary account number (PAN) for card payments, or a US or Canadian bank account number and routing number for an ACH bank account. An instrument identifier token can exist independently, or it can be associated with a payment instrument.

An instrument identifier token can also contain an associated network token.

Instrument identifier tokens are associated with these features:

Card Art

TMS card art helps your customers select a card. See Card Art (on page 344).

Enrollable Network Tokens

TMS can enroll certain *network tokens* in an instrument identifier token to be used for future payments. Future payments require only the instrument identifier token for the payment information. The types of network tokens you can enroll into an instrument identifier are tokens used for in-app payment methods such as:

- Android Pay
- Apple Pay
- Chase Pay
- Google Pay
- Samsung Pay
- Visa Click to Pay

See Create an Instrument Identifier for Enrollable Network Tokens (on page 248).

Push Provisioning

Push provisioning connects you with participating issuers to quickly provide credentials to your customers. See Push Provisioning for Network Tokens (on page 339).

Related information

Manage Instrument Identifier Tokens *(on page 243)*
Payments with Instrument Identifier Tokens *(on page 271)*

Manage Instrument Identifier Tokens

This section contains information on managing instrument identifier tokens.

The instrument identifier token type represents the tokenized Primary Account Number (PAN) for card payments, or US or Canadian bank account number and routing number. An instrument identifier can contain a credit card, ACH bank account, or tokenized card such as Apple Pay or Android Pay. You can create, retrieve, update, or delete an instrument identifier by submitting an HTTP `POST`, `GET`, `PATCH`, or `DELETE` operation to the `/tms/v1/instrumentidentifiers` endpoint.

Use the TMS REST API instrument identifier endpoint to:

- Create an instrument identifier token (on page 243)
- Retrieve an instrument identifier token (on page 253)
- Update an instrument identifier (on page 255)
- Delete an instrument identifier (on page 269)
- List payment instruments for an instrument identifier (on page 257)

For more information on instrument identifier tokens, see Instrument Identifier Tokens (on page 22).

Create an Instrument Identifier

This section shows you how to create an instrument identifier.

Endpoint

Test: `POST https://apitest.cybersource.com/tms/v1/instrumentidentifiers`

Production: `POST https://api.cybersource.com/tms/v1/instrumentidentifiers`

Production in India: `POST https://api.in.cybersource.com/tms/v1/instrumentidentifiers`

Required Fields for Creating an Instrument Identifier

card.number

Related Information

- API Field Reference for the REST API

Optional Fields for Creating an Instrument Identifier

bankAccount.number

bankAccount.routingNumber

billTo.address1

billTo.address2

billTo.administrativeArea

billTo.country

billTo.locality

billTo.postalCode

card.expirationMonth

card.expirationYear

card.securityCode

processingInformation.authorizationOptions.initiator.merchantInitiatedTransaction.previousTransactionID

Related Information

- API Field Reference for the REST API

REST Example: Creating a Card Instrument Identifier

Request

```
{
  "card": {
    "number": "4111XXXX11111111"
  }
}
```

Response to a Successful Request


```
{
  "_links": {
    "self": {
      "href":
 "https://apitest.cybersource.com/tms/v1/instrumentidentifiers/701000000001624111
11"
    },
    "paymentInstruments": {
      "href":
 "https://apitest.cybersource.com/tms/v1/instrumentidentifiers/701000000001624111
11/paymentinstruments"
    }
  },
  "id": "7010000000016241111",
  "object": "instrumentIdentifier",
  "state": "ACTIVE",
  "card": {
    "number": "411111XXXXXX1111"
  },
  "metadata": {
    "creator": "testrest"
  }
}
```

REST Example: Creating a Bank Account Instrument Identifier

Request

```
POST https://apitest.cybersource.com/tms/v1/instrumentidentifiers
{
  "bankAccount": {
    "number": "4100",
    "routingNumber": "X71923284"
  }
}
```

Response to a Successful Request

```
{
  "_links": {
    "self": {

  "href": "https://
apitest.cybersource.com/tms/v1/instrumentidentifiers/A7A91A2CA872B272E05340588D0A0
699"
    },
    "paymentInstruments": {

  "href": "https://
apitest.cybersource.com/tms/v1/instrumentidentifiers/A7A91A2CA872B272E05340588D0A0
699/paymentinstruments"
    }
  },
  "id": "A7A91A2CA872B272E05340588D0A0699",
  "object": "instrumentIdentifier",
  "state": "ACTIVE",
  "bankAccount": {
    "number": "XXXX",
    "routingNumber": "X71923284"
  },
  "metadata": {
    "creator": "testrest"
  }
}
```

Create an Instrument Identifier for Enrollable Network Tokens

Important: To create an instrument identifier for an enrollable network token using the Token Management Service (TMS), you must send the request using message-level encryption (MLE). For more information about MLE, see Message-Level Encryption Keys (on page 65).

TMS can enroll certain network tokens into an instrument identifier token for future payments. Any future payments will require only the instrument identifier token for the payment information.

Enrollable network tokens can be used for these in-app payment methods:

- Android Pay
- Apple Pay
- Chase Pay
- Google Pay
- Samsung Pay
- Visa Click to Pay

These tokenized payment methods are also referred to as *digital payments*, *digital wallets*, and *tokenized cards*.

Endpoint

Test: `POST https://apitest.cybersource.com/tms/v1/instrumentidentifiers`

Production: `POST https://api.cybersource.com/tms/v1/instrumentidentifiers`

Production in India: `POST https://api.in.cybersource.com/tms/v1/instrumentidentifiers`

Header

Set the component type in the request header to `application/jose`.

Response to a Successful Request

A successful response includes the instrument identifier in the **id** field and the `TOKEN` indicator in the **tokenizedCard.source** field. The `TOKEN` indicator denotes that the instrument identifier was created from a device token. A payment account reference (PAR) number is also returned in the **issuer.paymentAccountReference** field.

Cybersource returns a reason code in the **details.reason** response field to indicate the reason for an API request's status. For more information about all possible reason codes, see the *Cybersource Reason Codes with REST API response* article.

Merchant-Initiated Transactions

You can create an instrument identifier that stores a device token while you are requesting an authorization. Such requests are typically performed for follow-on merchant-initiated transactions. For more information about how to create an instrument identifier within an authorization request, see these sections in the *Payments Developer Guide*:

- Installment Payments
- Recurring Payments
- Unscheduled COF Payments

Required Fields for Creating an Instrument Identifier for a Device Token

card.number

type

Set to `enrollable token`.

Related Information

- API Field Reference for the REST API

Optional Fields for Creating an Instrument Identifier

bankAccount.number

bankAccount.routingNumber

billTo.address1

billTo.address2

billTo.administrativeArea

billTo.country

billTo.locality

billTo.postalCode

card.expirationMonth

card.expirationYear

card.securityCode

processingInformation.authorizationOptions.initiator.merchantInitiatedTransaction.previousTransactionID

Related Information

- API Field Reference for the REST API

REST Example: Creating an Instrument Identifier for a Device Token

Request

```
{
  "type": "enrollable token",
  "card": {
    "number": "41111XXXX1111111"
  }
}
```

Response to a Successful Request


```
{
  "_links": {
    "self": {
      "href":
 "https://apitest.cybersource.com/tms/v1/instrumentidentifiers/70300000000149115
15"
    },
    "paymentInstruments": {
      "href":
 "https://apitest.cybersource.com/tms/v1/instrumentidentifiers/70300000000149115
15/paymentinstruments"
    }
  },
   "id": "7040000000015027161",
  "object": "instrumentIdentifier",
  "state": "ACTIVE",
  "tokenizedCard": {
    "source": "TOKEN",
    "state": "ACTIVE",
    "enrollmentId": "da1fb810b1b3e01db5b215de5261df01",
    "tokenReferenceId": "090673c4811a91960f021ad3a24e2e01",
    "number": "41111XXXX1111111",
    "type": "visa",
    "card": {
      "suffix": "1111"
    }
  },
  "card": {
    "number": "41111XXXX1111111"
  },
  "issuer": {
    "paymentAccountReference": "V0010013024026372674590402581"
  },
  "metadata": {
    "creator": "testrest"
  }
}
```

Retrieve an Instrument Identifier

This section shows you how to retrieve an instrument identifier.

Endpoint

Test: `GET https://apitest.cybersource.com/tms/v1/instrumentidentifiers/{instrumentIdentifierTokenId}`

Production: `GET https://api.cybersource.com/tms/v1/instrumentidentifiers/{instrumentIdentifierTokenId}`

Production in India: `GET https://api.in.cybersource.com/tms/v1/instrumentidentifiers/{instrumentIdentifierTokenId}`

`{instrumentIdentifierTokenId}` is the instrument identifier token ID returned in the **id** field when you created the instrument identifier token. For more information, see Create an Instrument Identifier (on page 243).

REST Example: Retrieving an Instrument Identifier

Request

```
GET https://apitest.cybersource.com/tms/v1/instrumentidentifiers/7030000000046226
486
```

Response to a Successful Request


```
{
    "_links": {
        "self": {
            "href":
 https://apitest.visaacceptance.com/tms/v1/instrumentidentifiers/7030000000046226
486
        },
        "paymentInstruments": {
            "href":
 https://apitest.visaacceptance.com/tms/v1/instrumentidentifiers/70300000000462264
86/paymentinstruments
        }
    },
    "id": "7030000000046226486",
    "object": "instrumentIdentifier",
    "state": "ACTIVE",
    "tokenizedCard": {
        "state": "ACTIVE",
        "enrollmentId": "80efd04a838265dcdf9d114eb542be01",
        "tokenReferenceId": "003769ea947eb057c13c192f1d26f002",
        "number": "489537XXXXXX2655",
        "expirationMonth": "12",
        "expirationYear": "2030",
        "type": "visa",
        "card": {
            "suffix": "6486",
            "expirationMonth": "12",
            "expirationYear": "2030"
        }
    },
    "card": {
        "number": "489537XXXXXX6486"
    },
    "issuer": {
        "paymentAccountReference": "V0010013020037567013722096631"
    },
    "metadata": {
        "creator": "testacct"
    }
}
```

Update an Instrument Identifier

This section shows you how to update an instrument identifier.

Endpoint

Test: `PATCH https://apitest.cybersource.com/tms/v1/instrumentidentifiers/{instrumentIdentifierTokenId}`

Production: `PATCH https://api.cybersource.com/tms/v1/instrumentidentifiers/{instrumentIdentifierTokenId}`

Production in India: `PATCH https://api.in.cybersource.com/tms/v1/instrumentidentifiers/{instrumentIdentifierTokenId}`

`{instrumentIdentifierTokenId}` is the instrument identifier token ID returned in the **id** field when you created the instrument identifier token. For more information, see Create an Instrument Identifier (on page 243).

Optional Fields for Updating an Instrument Identifier

bankAccount.number

bankAccount.routingNumber

billTo.address1

billTo.address2

billTo.administrativeArea

billTo.country

billTo.locality

billTo.postalCode

card.expirationMonth

card.expirationYear

card.securityCode

Related Information

- API Field Reference for the REST API

REST Example: Updating an Instrument Identifier

Request

```
PATCH https://apitest.cybersource.com/tms/v1/instrumentidentifiers/7010000000016241111
```

Response to a Successful Request


```
{
  "_links": {
    "self": {
      "href":
 "https://apitest.cybersource.com/tms/v1/instrumentidentifiers/701000000001624111
11"
    },
    "paymentInstruments": {
      "href":
 "https://apitest.cybersource.com/tms/v1/instrumentidentifiers/701000000001624111
11/paymentinstruments"
    }
  },
  "id": "7010000000016241111",
  "object": "instrumentIdentifier",
  "state": "ACTIVE",
  "card": {
    "number": "411111XXXXXX1111"
  },
  "processingInformation": {
    "authorizationOptions": {
      "initiator": {
        "merchantInitiatedTransaction": {
          "previousTransactionId": "123456789012345"
        }
      }
    }
  },
  "metadata": {
    "creator": "testrest"
  }
}
```


Retrieve an Instrument Identifier's Payment Instruments

This section shows you how to retrieve the payment instrument tokens associated with an instrument identifier token.

Endpoint

Test: `GET https://apitest.cybersource.com/tms/v1/instrumentidentifiers/{instrumentIdentifierTokenId}/paymentinstruments?`

Production: `GET https://api.cybersource.com/tms/v1/instrumentidentifiers/{instrumentIdentifierTokenId}/paymentinstruments?`

Production in India: `GET https://api.in.cybersource.com/tms/v1/instrumentidentifiers/{instrumentIdentifierTokenId}/paymentinstruments?`

{instrumentIdentifierTokenId} is the instrument identifier token ID returned in the **id** field when you created the instrument identifier token. For more information, see Create an Instrument Identifier (on page 243).

Use these query string parameters to filter the list of payment instrument tokens:

- `offset` — Page offset number.
- `limit` — Maximum number of items you would like returned.

REST Example: Retrieving an Instrument Identifier's Payment Instruments

Request

```
GET https://apitest.cybersource.com/tms/v1/instrumentidentifiers/7010000000016241111/paymentinstruments?offset=0&limit=5
```

Response to a Successful Request


```
{
  "_links": {
    "self": {
      "href":
 "https://apitest.cybersource.com/tms/v1/instrumentidentifiers/70100000000162411
11/paymentinstruments?offset=0&limit=5"
    },
    "first": {
      "href":
 "https://apitest.cybersource.com/tms/v1/instrumentidentifiers/70100000000162411
11/paymentinstruments?offset=0&limit=5"
    },
    "next": {
      "href":
 "https://apitest.cybersource.com/tms/v1/instrumentidentifiers/70100000000162411
11/paymentinstruments?offset=5&limit=5"
    },
    "last": {
      "href":
 "https://apitest.cybersource.com/tms/v1/instrumentidentifiers/70100000000162411
11/paymentinstruments?offset=120820&limit=5"
    }
  },
  "object": "collection",
  "offset": 0,
  "limit": 5,
  "count": 5,
  "total": 120825,
  "_embedded": {
    "paymentInstruments": [
      {
        "_links": {
          "self": {
            "href":
 "https://apitest.cybersource.com/tms/v1/paymentinstruments/F396A4DD49CA23ADE053A2
598D0AECC4"
          },
          "customer": {
            "href":
 "https://apitest.cybersource.com/tms/v1/customers/F396A4DD49CB23ADE053A2598D0AEC
C4"
          }
        },
        "id": "F396A4DD49CA23ADE053A2598D0AECC4",
        "object": "paymentInstrument",
        "state": "ACTIVE",
        "card": {
```

```
          "expirationMonth": "12",
          "expirationYear": "2031",
          "type": "visa"
        },
        "buyerInformation": {
          "currency": "USD"
        },
        "billTo": {
          "firstName": "JOHN",
          "lastName": "DOE",
          "address1": "1 Market St",
          "locality": "san francisco",
          "administrativeArea": "CA",
          "postalCode": "94105",
          "country": "US",
          "email": "test@cybs.com",
          "phoneNumber": "4158880000"
        },
        "processingInformation": {
          "billPaymentProgramEnabled": false
        },
        "metadata": {
          "creator": "testrest"
        },
        "_embedded": {
          "instrumentIdentifier": {
            "_links": {
              "self": {
                "href":
 "https://apitest.cybersource.com/tms/v1/instrumentidentifiers/70100000000162411
11"
              },
              "paymentInstruments": {
                "href":
 "https://apitest.cybersource.com/tms/v1/instrumentidentifiers/70100000000162411
11/paymentinstruments"
              }
            },
            "id": "7010000000016241111",
            "object": "instrumentIdentifier",
            "state": "ACTIVE",
            "card": {
              "number": "411111XXXXXX1111"
            },
            "processingInformation": {
              "authorizationOptions": {
                "initiator": {
                  "merchantInitiatedTransaction": {
                    "previousTransactionId": "123456789012345"
```

```
                }
              }
            }
          },
          "metadata": {
            "creator": "testrest"
          }
        }
      }
    },
    {
      "_links": {
        "self": {
          "href":
 "https://apitest.cybersource.com/tms/v1/paymentinstruments/F3969009C44DED0DE053AF
598E0AD4E0"
        },
        "customer": {
          "href":
 "https://apitest.cybersource.com/tms/v1/customers/F396A109D27377A5E053AF598E0AA3
4A"
        }
      },
      "id": "F3969009C44DED0DE053AF598E0AD4E0",
      "object": "paymentInstrument",
      "state": "ACTIVE",
      "card": {
        "type": "visa"
      },
      "metadata": {
        "creator": "testrest"
      },
      "_embedded": {
        "instrumentIdentifier": {
          "_links": {
            "self": {
              "href":
 "https://apitest.cybersource.com/tms/v1/instrumentidentifiers/70100000000162411
11"
            },
            "paymentInstruments": {
              "href":
 "https://apitest.cybersource.com/tms/v1/instrumentidentifiers/70100000000162411
11/paymentinstruments"
            }
          },
          "id": "7010000000016241111",
          "object": "instrumentIdentifier",
          "state": "ACTIVE",
```

```
            "card": {
              "number": "411111XXXXXX1111"
            },
            "processingInformation": {
              "authorizationOptions": {
                "initiator": {
                  "merchantInitiatedTransaction": {
                    "previousTransactionId": "123456789012345"
                  }
                }
              }
            },
            "metadata": {
              "creator": "testrest"
            }
          }
        }
      },
      {
        "_links": {
          "self": {
            "href":
 "https://apitest.cybersource.com/tms/v1/paymentinstruments/F396A109F3637776E053AF
598E0A87E4"
          },
          "customer": {
            "href":
 "https://apitest.cybersource.com/tms/v1/customers/F396A109D27377A5E053AF598E0AA3
4A"
          }
        },
        "id": "F396A109F3637776E053AF598E0A87E4",
        "object": "paymentInstrument",
        "state": "ACTIVE",
        "card": {
          "expirationMonth": "12",
          "expirationYear": "2031",
          "type": "visa"
        },
        "billTo": {
          "firstName": "John",
          "lastName": "Doe",
          "company": "Company Name",
          "address1": "1 Market St",
          "locality": "San Francisco",
          "administrativeArea": "CA",
          "postalCode": "94105",
          "country": "US",
          "email": "test@cybs.com",
```

```
          "phoneNumber": "4158880000"
        },
        "metadata": {
          "creator": "testrest"
        },
        "_embedded": {
          "instrumentIdentifier": {
            "_links": {
              "self": {
                "href":
 "https://apitest.cybersource.com/tms/v1/instrumentidentifiers/70100000000162411
11"
              },
              "paymentInstruments": {
                "href":
 "https://apitest.cybersource.com/tms/v1/instrumentidentifiers/70100000000162411
11/paymentinstruments"
              }
            },
            "id": "7010000000016241111",
            "object": "instrumentIdentifier",
            "state": "ACTIVE",
            "card": {
              "number": "411111XXXXXX1111"
            },
            "processingInformation": {
              "authorizationOptions": {
                "initiator": {
                  "merchantInitiatedTransaction": {
                    "previousTransactionId": "123456789012345"
                  }
                }
              }
            },
            "metadata": {
              "creator": "testrest"
            }
          }
        }
      },
      {
        "_links": {
          "self": {
            "href":
 "https://apitest.cybersource.com/tms/v1/paymentinstruments/F3965253C47640F5E053AF
598E0AA05A"
          }
        },
        "id": "F3965253C47640F5E053AF598E0AA05A",
```

```
        "object": "paymentInstrument",
        "state": "ACTIVE",
        "card": {
          "expirationMonth": "02",
          "expirationYear": "2028",
          "type": "visa"
        },
        "billTo": {
          "firstName": "John",
          "lastName": "Snow"
        },
        "metadata": {
          "creator": "testrest"
        },
        "_embedded": {
          "instrumentIdentifier": {
            "_links": {
              "self": {
                "href":
 "https://apitest.cybersource.com/tms/v1/instrumentidentifiers/70100000000162411
11"
              },
              "paymentInstruments": {
                "href":
 "https://apitest.cybersource.com/tms/v1/instrumentidentifiers/70100000000162411
11/paymentinstruments"
              }
            },
            "id": "7010000000016241111",
            "object": "instrumentIdentifier",
            "state": "ACTIVE",
            "card": {
              "number": "411111XXXXXX1111"
            },
            "processingInformation": {
              "authorizationOptions": {
                "initiator": {
                  "merchantInitiatedTransaction": {
                    "previousTransactionId": "123456789012345"
                  }
                }
              }
            },
            "metadata": {
              "creator": "testrest"
            }
          }
        }
      },
```

```
      {
        "_links": {
          "self": {
            "href":
 "https://apitest.cybersource.com/tms/v1/paymentinstruments/F395F6426D9A30AEE053AF
598E0A5BD4"
          }
        },
        "id": "F395F6426D9A30AEE053AF598E0A5BD4",
        "object": "paymentInstrument",
        "state": "ACTIVE",
        "card": {
          "expirationMonth": "12",
          "expirationYear": "2031",
          "type": "visa"
        },
        "billTo": {
          "firstName": "John",
          "lastName": "Doe",
          "company": "Company Name",
          "address1": "1 Market St",
          "locality": "San Francisco",
          "administrativeArea": "CA",
          "postalCode": "94105",
          "country": "US",
          "email": "test@cybs.com",
          "phoneNumber": "4158880000"
        },
        "metadata": {
          "creator": "testrest"
        },
        "_embedded": {
          "instrumentIdentifier": {
            "_links": {
              "self": {
                "href":
 "https://apitest.cybersource.com/tms/v1/instrumentidentifiers/70100000000162411
11"
              },
              "paymentInstruments": {
                "href":
 "https://apitest.cybersource.com/tms/v1/instrumentidentifiers/70100000000162411
11/paymentinstruments"
              }
            },
            "id": "7010000000016241111",
            "object": "instrumentIdentifier",
            "state": "ACTIVE",
            "card": {
```

```
            "number": "411111XXXXXX1111"
          },
          "processingInformation": {
            "authorizationOptions": {
              "initiator": {
                "merchantInitiatedTransaction": {
                  "previousTransactionId": "123456789012345"
                }
              }
            }
          },
          "metadata": {
            "creator": "testrest"
          }
        }
      }
    ]
  }
}
```

Retrieve an Instrument Identifier with an Unmasked Card Number

This section shows you how retrieve an instrument identifier with an unmasked card number.

Important: To retrieve unmasked payment details, you must ensure that your MLE key pair and your token vault are configured correctly. For more information on MLE keys, see Message-Level Encryption Keys (on page 65). For more information on token vaults, see Token Vault Management (on page 62). If necessary, contact your Cybersource account manager or customer support.

The response is BASE 64-encoded JSON web encryption (JWE) token. The decoded JWE has these elements:

```
{ "alg": "RSA-OAEP-256", //The algorithm used to encrypt the CEK
    "cty": "json", //The content type
    "typ": "JWT", //The token type
    "enc": "A256GCM", //The algorithm that is used to encrypt the message
    "kid": "keyId" //The serial number of shared public cert for encryption of CEK
}
<Encrypted Data> //The encrypted payload that matches the JSON response normally
 returned by the TMS API, except with an unmasked payment details
```

Header Configuration

You must pass this request header to retrieve unmasked payment details: `Accept: application/jose`.

The term `application/jose` refers to Javascript Object Signing and Encryption (JOSE). JOSE is a framework that provides end-to-end security to JavaScript Object Notation (JSON)-based data structures. JOSE achieves this by offering a collection of specifications to encrypt and digitally sign JSON payloads. In this case, the response is message-level encrypted using a JSON Web Token (JWT).

Endpoint

Test: `GET https://apitest.cybersource.com/tms/v1/instrumentidentifiers/{instrumentIdentifierTokenId}`

Production: `GET https://api.cybersource.com/tms/v1/instrumentidentifiers/{instrumentIdentifierTokenId}`

Production in India: `GET https://api.in.cybersource.com/tms/v1/instrumentidentifiers/{instrumentIdentifierTokenId}`

{instrumentIdentifierTokenId} is the instrument identifier token ID returned in the **id** field when you created the instrument identifier token. For more information, see Create an Instrument Identifier (on page 243).

REST Example: Retrieving an Instrument Identifier with an Unmasked Card Number

Request

```
GET https://apitest.cybersource.com/tms/v1/instrumentidentifiers/7010000000016241
111
```

Response to a Successful Request


```
eyJraWQiOiJiYTE1ZDRmMTIzMTM0NjlkZjg5MDM1Nzk2YWE4Nzc4ZGM0NTY4ODlkIiwiY3R5IjoianNvbi
IsInR5cCI6IkpXVCIsImVuYyI6IkEyNTZHQ00iLCJhbGciOiJSU0EtT0FFUC0yNTYifQ.N_XRPaRJACKNH
BAUvXIu11eoB
 kgouHb5mrA1LL-WHVfKRpfUoGpHRVp0WRy7b1NLh4-qAlLI3QKnxxplx4tzSaJCn3kQDNt0BnRmKecRvF
TGKXk09eATF8J7lLfNjYZEgZgA4qe3FdIEWIMN_BwQMJMEy0cMJdpyGtvUt9G6rgmQUDsjwSDU5tQNMopj
gqjDUw6rBjbxTprNtBLpNCqjbSe4-vW_xiZIfFpQs_45YPWV4fRn5YuH7ebfckB1evTdfGRlBMHXfjac_Q
A8a1gMv_50T2y1VXllam2OSC2hSabOtd43pGDsFwj5HhOGjobb6GprbNedlIBL5Mlo-2_wCg.OkTe1Z7Or
edhIrF_.3eEeC9OUfz7uXxl1FLSZZNFGUiX7vk77SGVCW7cypDuVpy5QpK2wVJzTYrjJFgGlEiE05GwXP0
4gOsOOp5C6OEhCXKbdGMZO_V0FAyxk1dnx28ur-cSG-86HdbBRbWsvcuh4ghMqx8WTlA-M13YKubY6L2Lc
K0yWROn9MrYlUWzgJFjXFZDkpCxsHpMtvXRxcF6nTQkJD4rw_SHGuHqWbVlQKyIEBcvbuyecjYtz7iZtP_
HS349TOOmpbJDxJ-X8exZy3LLTmD7PHjpySYGx-svlkP-Qu4yi_xFtzmkwf7T7O56SAa9DidDeH9ftGi7V
67MBMBGK6Ndl8nK4sn6SieBDMWxnFthNdHZFEhlSONIywGfE-mYI5nuagrNVOo-ZQqJ2woYXdocdEvyTQ7
oDvRy432872l6nUDTZcVdYlVj79KDrW73LjvUYWcAvXZr0bDgI-e1YNziInqgi3DlNNeL6W2srYrSuqJG5
-NnWIISt3Pb8qfa2ve06uRhztpyWisWEZOCVG1SLg_LZTPjaDoe2woJ1kyP2VaEM4VoRynQ0dCZsLlpu8_
s24rj96T-qoi2QkUybUQ3rpYiUUPl1-jhhimMpar4wsJJRIsVfsf6KVz876ReMgvW1Jzm5G0Ypj7acvvqn
DAeMEfRzXvpLvAVpGXP6RbVXuyg3wyUg_8-PqOlllRiavS8eg9-ZdeuAkPQ.4vZxOPrGjw51SFJmn_cF3g
```


Delete an Instrument Identifier

This section shows you how to delete an instrument identifier.

Endpoint

Test: `DELETE https://apitest.cybersource.com/tms/v1/instrumentidentifiers/{instrumentIdentifierTokenId}`

Production: `DELETE https://api.cybersource.com/tms/v1/instrumentidentifiers/{instrumentIdentifierTokenId}`

Production in India: `DELETE https://api.in.cybersource.com/tms/v1/instrumentidentifiers/{instrumentIdentifierTokenId}`

{instrumentIdentifierTokenId} is the instrument identifier token ID returned in the **id** field in the **id** field when you created the instrument identifier token. For more information, see Create an Instrument Identifier (on page 243).

REST Example: Deleting an Instrument Identifier

Request

```
DELETE https://apitest.cybersource.com/tms/v1/instrumentidentifiers/701000000001624111
```

Response to a Successful Request

A successful delete response returns an empty `HTTP 204 No Content` status. For more information, see HTTP Status Codes (on page 358).

Payments with Instrument Identifier Tokens

This section contains information on making payments using instrument identifier tokens.

An instrument identifier token represents either a payment card number or, in the case of an ACH bank account, the routing and account numbers. The expiration date and billing address fields are pass through fields. The pass-through fields are used for payment network token enrollment with card associations.

You can make a payment using an existing instrument identifier token or create one. To make a payment using a new instrument identifier token, you must include token creation in the authorization request. For example:

- Create an Instrument Identifier Token with Validated Payment Details (on page 272)

To process a payment using an existing instrument identifier token, you must include the instrument identifier token ID as the value in the `paymentInformation.instrumentIdentifier.id` field. For example:

- Authorize a Payment with an Instrument Identifier (on page 277)
- Making a Credit with an Instrument Identifier (on page 284)

For more information on instrument identifier tokens, see Instrument Identifier Tokens (on page 22).

Create an Instrument Identifier Token with Validated Payment Details

This section shows you how to create a instrument identifier token with validated payment details.

Endpoint

Test: `POST https://apitest.cybersource.com/pts/v2/payments`

Production: `POST https://api.cybersource.com/pts/v2/payments`

Production in India: `POST https://api.in.cybersource.com/pts/v2/payments`

`customerTokenId` is the customer tokenID returned in the **id** field when you created the customer token. For more information, see Create a Customer (on page 92).

Required Fields for Creating an Instrument Identifier Token with Validated Payment Details

orderInformation.amountDetails.currency

orderInformation.amountDetails.totalAmount

orderInformation.billTo.address1

orderInformation.billTo.administrativeArea

orderInformation.billTo.country

orderInformation.billTo.email

orderInformation.billTo.firstName

orderInformation.billTo.lastName

orderInformation.billTo.locality

orderInformation.billTo.postalCode

paymentInformation.card.expirationMonth

paymentInformation.card.expirationYear

paymentInformation.card.number

paymentInformation.card.type

processingInformation.actionList

Set the value to `TOKEN_CREATE`.

processingInformation.actionTokenTypes

Set the value to `instrumentIdentifier`.

Related Information

- API Field Reference for the REST API

REST Example: Creating an Instrument Identifier with Validated Payment Details

Request

```
{
    "clientReferenceInformation": {
        "code": "TC50171_3"
    },
    "processingInformation": {
        "commerceIndicator": "internet",
        "actionList": [
            "TOKEN_CREATE"
        ],
        "actionTokenTypes": [
            "instrumentIdentifier"
        ]
    },
    "orderInformation": {
        "billTo": {
            "country": "US",
            "lastName": "Deo",
            "address2": "Address 2",
            "address1": "201 S. Division St.",
            "postalCode": "48104-2201",
            "locality": "Ann Arbor",
            "administrativeArea": "MI",
            "firstName": "John",
            "phoneNumber": "999999999",
            "district": "MI",
            "buildingNumber": "123",
            "company": "Visa",
            "email": "test@cybs.com"
        },
```

```
        "shipTo": {
            "country": "US",
            "lastName": "Deo",
            "address2": "Address 2",
            "address1": "201 S. Division St.",
            "postalCode": "48104-2201",
            "locality": "Ann Arbor",
            "administrativeArea": "MI",
            "firstName": "John",
            "phoneNumber": "999999999",
            "district": "MI",
            "buildingNumber": "123",
            "company": "Visa",
            "email": "test@cybs.com"
        },
        "amountDetails": {
            "totalAmount": "102.00",
            "currency": "USD"
        }
    },
    "paymentInformation": {
        "card": {
            "expirationYear": "2031",
            "number": "4895379987X11515",
            "securityCode": "890",
            "expirationMonth": "12",
            "type": "001"
        }
    }
}
```

Response to a Successful Request


```
{
    "_links": {
        "authReversal": {
            "method": "POST",
            "href": "/pts/v2/payments/676063487034615490 3955/reversals"
        },
        "self": {
            "method": "GET",
            "href": "/pts/v2/payments/6760634870346154903955"
        },
        "capture": {
            "method": "POST",
            "href": "/pts/v2/payments/676063487034615490 3955/captures"
        }
    },
    "clientReferenceInformation": {
        "code": "TC50171_3"
    },
    "id": "6760634870346154903955",
    "orderInformation": {
        "amountDetails": {
            "authorizedAmount": "102.00",
            "currency": "USD"
        }
    },
    "paymentAccountInformation": {
        "card": {
            "type": "001"
        }
    },
    "paymentInformation": {
        "tokenizedCard": {
            "type": "001"
        },
        "card": {
            "type": "001"
        }
    },
    "pointOfSaleInformation": {
        "terminalId": "111111"
    },
    "processorInformation": {
        "paymentAccountReferenceNumber": "V0010013019326121174070050420",
        "approvalCode": "888888",
        "networkTransactionId": "123456789619999",
        "transactionId": "123456789619999",
        "responseCode": "100",
```

```
        "avs": {
            "code": "X",
            "codeRaw": "I1"
        }
    },
    "reconciliationId": "698162504DTIATR3",
    "status": "AUTHORIZED",
    "submitTimeUtc": "2023-02-10T21:11:27Z",
    "tokenInformation": {
        "instrumentidentifierNew": false,
        "instrumentIdentifier": {
            "state": "ACTIVE",
            "id": "7030000000014911515"
        }
    }
}
```

Authorize a Payment with an Instrument Identifier

This section provides the information you need in order to authorize a payment with an instrument identifier token.

Endpoint

Production: `POST https://api.cybersource.com/pts/v2/payments`

Test: `POST https://apitest.cybersource.com/pts/v2/payments`

Production in India: `POST https://api.in.cybersource.com/pts/v2/payments`

Required Fields for Authorizing a Payment with an Instrument Identifier

clientReferenceInformation.code

orderInformation.amountDetails.currency

orderInformation.amountDetails.totalAmount

paymentInformation.instrumentIdentifier.id

Set to the ID of the instrument identifier token you want to use.

Related Information

- API field reference guide for the REST API

REST Example: Authorizing a Payment with an Instrument Identifier

Request

```
{
  "clientReferenceInformation": {
    "code": "12345678"
  },
    "paymentInformation": {
        "instrumentIdentifier": {
            "id": "7010000000016241111"
        }
```

```
    },
    "orderInformation": {
        "amountDetails": {
            "currency": "USD",
            "totalAmount": "10.00"
        }
    }
}
```

Response to a Successful Request


```
{
  "_links": {
    "authReversal": {
      "method": "POST",
      "href": "/pts/v2/payments/705595528818605340495 3/reversals"
    },
    "self": {
      "method": "GET",
      "href": "/pts/v2/payments/705595528818605340495 3"
    },
    "capture": {
      "method": "POST",
      "href": "/pts/v2/payments/705595528818605340495 3/captures"
    }
  },
  "clientReferenceInformation": {
    "code": "12345678"
  },
  "id": "705595528818605340495 3",
  "orderInformation": {
    "amountDetails": {
      "authorizedAmount": "10.00",
      "currency": "USD"
    }
  },
  "paymentAccountInformation": {
    "card": {
      "type": "001"
    }
  },
  "paymentInformation": {
    "tokenizedCard": {
      "type": "001"
    },
    "instrumentIdentifier": {
      "id": "7010000000016241111",
      "state": "ACTIVE"
    },
    "card": {
      "type": "001"
    }
  },
  "pointOfSaleInformation": {
    "terminalId": "111111"
  },
  "processorInformation": {
    "approvalCode": "888888",
```

```
    "networkTransactionId": "123456789619999",
    "transactionId": "123456789619999",
    "responseCode": "100",
    "avs": {
      "code": "1"
    }
  },
  "reconciliationId": "67468271CRIL0U24",
  "status": "AUTHORIZED",
  "submitTimeUtc": "2024-01-18T16:32:09Z"
}
```

REST Example: Authorizing a Payment with an Instrument Identifier While Creating TMS Tokens

Request

```
{
  "clientReferenceInformation": {
    "code": "TC50171_3"
  },
  "processingInformation": {
    "actionList": [
      "TOKEN_CREATE"
    ],
    "actionTokenTypes": [
      "customer",
      "paymentInstrument",
      "shippingAddress"
    ]
  },
  "paymentInformation": {
    "instrumentIdentifier": {
      "id": "7010000000016241111"
    }
  },
  "orderInformation": {
    "amountDetails": {
      "totalAmount": "102.21",
      "currency": "USD"
    },
    "billTo": {
      "firstName": "John",
      "lastName": "Doe",
      "address1": "1 Market St",
```

```
      "locality": "san francisco",
      "administrativeArea": "CA",
      "postalCode": "94105",
      "country": "US",
      "email": "test@cybs.com",
      "phoneNumber": "4158880000"
    },
    "shipTo": {
      "firstName": "John",
      "lastName": "Doe",
      "address1": "1 Market St",
      "locality": "san francisco",
      "administrativeArea": "CA",
      "postalCode": "94105",
      "country": "US"
    }
  }
}
```

Response to a Successful Request


```
{
  "_links": {
    "authReversal": {
      "method": "POST",
      "href": "/pts/v2/payments/711467984037668720395 5/reversals"
    },
    "self": {
      "method": "GET",
      "href": "/pts/v2/payments/71146798403766872039 55"
    },
    "capture": {
      "method": "POST",
      "href": "/pts/v2/payments/711467984037668720395 5/captures"
    }
  },
  "clientReferenceInformation": {
    "code": "TC50171_3"
  },
  "id": "7114679840376687203955",
  "orderInformation": {
    "amountDetails": {
      "authorizedAmount": "102.21",
      "currency": "USD"
    }
  },
  "paymentAccountInformation": {
    "card": {
      "type": "001"
    }
  },
  "paymentInformation": {
    "tokenizedCard": {
      "type": "001"
    },
    "instrumentIdentifier": {
      "id": "7010000000016241111",
      "state": "ACTIVE"
    },
    "card": {
      "type": "001"
    }
  },
  "pointOfSaleInformation": {
    "terminalId": "111111"
  },
  "processorInformation": {
    "approvalCode": "888888",
```

```
    "networkTransactionId": "123456789619999",
    "transactionId": "123456789619999",
    "responseCode": "100",
    "avs": {
      "code": "X",
      "codeRaw": "I1"
    }
  },
  "reconciliationId": "623971212U7PN4IU",
  "status": "AUTHORIZED",
  "submitTimeUtc": "2024-03-26T15:46:24Z",
  "tokenInformation": {
    "shippingAddress": {
      "id": "14930C904FC4D97BE063A2598D0AE0F1"
    },
    "paymentInstrument": {
      "id": "149310A4A924E911E063A2598D0A47AD"
    },
    "customer": {
      "id": "14930C904FC1D97BE063A2598D0AE0F1"
    }
  }
}
```

Making a Credit with an Instrument Identifier

This section shows you how to make a credit with an instrument identifier token.

Endpoint

Test: `POST https://apitest.cybersource.com/pts/v2/credits`

Production: `POST https://api.cybersource.compts/v2/credits`

Production in India: `POST https://api.in.cybersource.compts/v2/credits`

Required Fields for Making a Credit with an Instrument Identifier

clientReferenceInformation.code

orderInformation.amountDetails.currency

orderInformation.amountDetails.totalAmount

paymentInformation.paymentInstrument.id

Set to the ID of the payment instrument token you want to use.

Related Information

- API field reference guide for the REST API

REST Example: Making a Credit with an Instrument Identifier

Request

```
{
  "clientReferenceInformation": {
    "code": "12345678"
  },
    "paymentInformation": {
        "instrumentIdentifier": {
            "id": "7010000000016241111"
        }
    },
    "orderInformation": {
```

```
        "amountDetails": {
            "currency": "USD",
            "totalAmount": "10.00"
        }
    }
}
```

Response to a Successful Request


```
{
  "_links": {
    "void": {
      "method": "POST",
      "href": "/pts/v2/credits/705597026106621240495 1/voids"
    },
    "self": {
      "method": "GET",
      "href": "/pts/v2/credits/705597026106621240495 1"
    }
  },
  "clientReferenceInformation": {
    "code": "12345678"
  },
  "creditAmountDetails": {
    "currency": "USD",
    "creditAmount": "10.00"
  },
  "id": "705597026106621240495 1",
  "orderInformation": {
    "amountDetails": {
      "currency": "USD"
    }
  },
  "paymentAccountInformation": {
    "card": {
      "type": "001"
    }
  },
  "paymentInformation": {
    "tokenizedCard": {
      "type": "001"
    },
    "instrumentIdentifier": {
      "id": "7010000000016241111",
      "state": "ACTIVE"
    },
    "card": {
      "type": "001"
    }
  },
  "processorInformation": {
    "approvalCode": "888888",
    "responseCode": "100"
  },
  "reconciliationId": "67445198PRILCQCQ",
```

```
  "status": "PENDING",
  "submitTimeUtc": "2024-01-18T16:57:06Z"
}
```

Legacy Tokens

A legacy token refers to an existing customer token, instrument identifier token, payment instrument token, or a shipping address token.

Payments with Legacy Tokens

This section contains information on making payments with legacy tokens.

It is recommended to use the TMS customer, payment instrument, or instrument identifier token ID fields rather than the legacy token ID. To process a payment using legacy token, instrument identifier token, or payment instrument token, you must include the legacy token ID in the request. For example:

- Authorizing a Payment with a Legacy Token (on page 289)
- Making a Credit with a Legacy Token (on page 293)

Authorizing a Payment with a Legacy Token

This section shows you how to authorize a payment with a legacy token.

Endpoint

Production: `POST https://api.cybersource.com/pts/v2/payments`

Test: `POST https://apitest.cybersource.com/pts/v2/payments`

Required Fields for Authorizing a Payment with a Legacy Token

clientReferenceInformation.code

paymentInformation.legacyToken.id

Include the ID of the legacy token you want to use to authorize a payment.

orderInformation.amountDetails.currency

orderInformation.amountDetails.totalAmount

Related Information

- API field reference guide for the REST API

REST Example: Authorizing a Payment with a Legacy Token

Request

```
{
  "clientReferenceInformation": {
    "code": "12345678"
  },
  "paymentInformation": {
    "legacyToken": {
      "id": "B21E6717A6F03479E05341588E0A303F"
    }
  },
  "orderInformation": {
    "amountDetails": {
      "totalAmount": "22.00",
```

```
      "currency": "USD"
    }
  }
}
```

Response to a Successful Request


```
{
  "_links": {
    "authReversal": {
      "method": "POST",
      "href": "/pts/v2/payments/705595634247678900495 1/reversals"
    },
    "self": {
      "method": "GET",
      "href": "/pts/v2/payments/705595634247678900495 1"
    },
    "capture": {
      "method": "POST",
      "href": "/pts/v2/payments/705595634247678900495 1/captures"
    }
  },
  "clientReferenceInformation": {
    "code": "12345678"
  },
  "id": "7055956342476789004951",
  "orderInformation": {
    "amountDetails": {
      "authorizedAmount": "22.00",
      "currency": "USD"
    }
  },
  "paymentAccountInformation": {
    "card": {
      "type": "001"
    }
  },
  "paymentInformation": {
    "tokenizedCard": {
      "type": "001"
    },
    "card": {
      "type": "001"
    }
  },
  "pointOfSaleInformation": {
    "terminalId": "111111"
  },
  "processorInformation": {
    "approvalCode": "888888",
    "networkTransactionId": "123456789619999",
    "transactionId": "123456789619999",
    "responseCode": "100",
    "avs": {
```

```
      "code": "X",
      "codeRaw": "I1"
    }
  },
  "reconciliationId": "67468431FRIIS246",
  "status": "AUTHORIZED",
  "submitTimeUtc": "2024-01-18T16:33:54Z"
}
```

Making a Credit with a Legacy Token

This section shows you how to make a credit with a legacy token.

Endpoint

Test: `POST https://apitest.cybersource.com/pts/v2/credits`

Production: `POST https://api.cybersource.com/pts/v2/credits`

Production in India: `POST https://api.in.cybersource.com/pts/v2/credits`

Required Fields for Making a Credit with a Legacy Token

clientReferenceInformation.code

orderInformation.amountDetails.currency

orderInformation.amountDetails.totalAmount

paymentInformation.legacyToken.id

Include the ID of the legacy token that you want to use to authorize a payment.

Related Information

- API field reference guide for the REST API

REST Example: Making a Credit with a Legacy Token

Request

```
{
  "clientReferenceInformation": {
    "code": "12345678"
  },
  "paymentInformation": {
    "legacyToken": {
      "id": "B21E6717A6F03479E05341588E0A303F"
    }
  },
  "orderInformation": {
```

```
    "amountDetails": {
      "totalAmount": "22.00",
      "currency": "USD"
    }
  }
}
```

Response to a Successful Request


```
{
  "_links": {
    "void": {
      "method": "POST",
      "href": "/pts/v2/credits/705597056209650970495 3/voids"
    },
    "self": {
      "method": "GET",
      "href": "/pts/v2/credits/705597056209650970495 3"
    }
  },
  "clientReferenceInformation": {
    "code": "12345678"
  },
  "creditAmountDetails": {
    "currency": "USD",
    "creditAmount": "22.00"
  },
  "id": "705597056209650970495 3",
  "orderInformation": {
    "amountDetails": {
      "currency": "USD"
    }
  },
  "paymentAccountInformation": {
    "card": {
      "type": "001"
    }
  },
  "paymentInformation": {
    "tokenizedCard": {
      "type": "001"
    },
    "card": {
      "type": "001"
    }
  },
  "processorInformation": {
    "approvalCode": "888888",
    "responseCode": "100"
  },
  "reconciliationId": "67444779FRILJT84",
```

```
  "status": "PENDING",
  "submitTimeUtc": "2024-01-18T16:57:36Z"
}
```

Network Tokens for Merchants

Network tokens are network scheme generated tokens that represent customer card information for secure transactions. Network tokens can be provisioned directly through Cybersource for merchants.

Payments with Network Tokens for Merchants

This section contains information on making payments with network tokens for merchants.

Network tokens are generated by the network scheme and represent customer card information for secure transactions. Network schemes include but are not limited to Visa, Mastercard, and American Express. Network tokens are mapped to instrument identifier tokens. The minimum card data required to request a network token is the PAN and the expiration date.

Payments are made using network tokens by default for merchants. To authorize a payment ignoring the network token, you must indicate this in the request. For example:

- Authorize a Payment While Ignoring Network Token (on page 298)

For more information on network tokens, see Network Tokens (on page 23).

Authorize a Payment While Ignoring Network Token

This section shows you how to authorize a payment ignoring a network token.

Endpoint

Test: `POST https://apitest.cybersource.com/pts/v2/payments`

Production: `POST https://api.cybersource.com/pts/v2/payments`

Production in India: `POST https://api.in.cybersource.com/pts/v2/payments`

Required Fields for Authorizing a Payment While Ignoring Network Token

clientReferenceInformation.code

paymentInformation.customer.id

paymentInformation.paymentInformation.id

paymentInformation.shippingAddress.id

orderInformation.amountDetails.currency

orderInformation.amountDetails.totalAmount

processingInformation.capture

processingInformation.commerceIndicator

tokenInformation.networkTokenOption

Set value to `ignore`.

Related Information

- API Field Reference for the REST API

REST Example: Authorizing a Payment While Ignoring Network Token

Request

```
{
    "clientReferenceInformation": {
        "code": "RTS-Auth"
    },
    "paymentInformation": {
        "card": {
            "expirationYear": "2031",
            "expirationMonth": "12",
            "type": "001"
        },
        "instrumentIdentifier": {
            "id": "7010000000016241111"
        }
    },
    "orderInformation": {
        "amountDetails": {
            "currency": "USD",
            "totalAmount": "1.00"
        }
    },
    "processingInformation": {
        "capture": "false",
        "commerceIndicator": "internet"
    },
    "tokenInformation": {
        "networkTokenOption": "ignore"
    }
}
```

Response to a Successful Request


```
{
    "_links": {
        "authReversal": {
            "method": "POST",
            "href": "/pts/v2/payments/676991344316641260495 1/reversals"
        },
        "self": {
            "method": "GET",
            "href": "/pts/v2/payments/6769913443166412604951"
        },
        "capture": {
            "method": "POST",
            "href": "/pts/v2/payments/6769913443166412604951/captures"
        }
    },
    "clientReferenceInformation": {
        "code": "RTS-Auth"
    },
    "id": "6769913443166412604951",
    "orderInformation": {
        "amountDetails": {
            "authorizedAmount": "1.00",
            "currency": "USD"
        }
    },
    "paymentAccountInformation": {
        "card": {
            "type": "001"
        }
    },
    "paymentInformation": {
        "tokenizedCard": {
            "type": "001"
        },
        "instrumentIdentifier": {
            "id": "7030000000014911515",
            "state": "ACTIVE"
        },
        "shippingAddress": {
            "id": "F537CE8DBA2F032CE053AF598E0A64F2"
        },
        "paymentInstrument": {
            "id": "F537E3D12322416EE053AF598E0AD771"
        },
        "card": {
            "type": "001"
        },
```

```
        "customer": {
            "id": "F537CE8DBA2C032CE053AF598E0A64F2"
        }
    },
    "pointOfSaleInformation": {
        "terminalId": "111111"
    },
    "processorInformation": {
        "paymentAccountReferenceNumber": "V0010013019326121174070050420",
        "approvalCode": "888888",
        "networkTransactionId": "123456789619999",
        "transactionId": "123456789619999",
        "responseCode": "100",
        "avs": {
            "code": "X",
            "codeRaw": "I1"
        }
    },
    "reconciliationId": "744295942E2LY3F8",
    "status": "AUTHORIZED",
    "submitTimeUtc": "2023-02-21T14:55:44Z"
}
```

Manage Merchant-Initiated Transactions and Network Tokens

This section contains information for using network tokens in merchant-initiated transactions (MIT).

Network tokens are generated by the network scheme and represent customer card information for secure transactions. Network schemes include but are not limited to Visa, Mastercard, and American Express. Network tokens are mapped to instrument identifier tokens. The minimum card data required to request a network token is the PAN and the expiration date.

You can manage merchant-initiated transactions using network tokens by including the relevant MIT information in requests sent to `tms/v2/customers`. For example:

- Update Merchant-Initiated Transaction Authorization Options (on page 302)

For more information on network tokens and credentialed transactions, see Network Tokens (on page 23) and Credentialed Transactions, respectively.

Update Merchant-Initiated Transaction Authorization Options

This section shows you how to update merchant-initiated transaction (MIT) authorization options.

Endpoint

Test: `PATCH https://apitest.cybersource.com/tms/v1/instrumentidentifiers/{instrumentIdentifierTokenId}`

Production: `PATCH https://api.cybersource.com/tms/v1/instrumentidentifiers/{instrumentIdentifierTokenId}`

Production in India: `PATCH https://api.in.cybersource.com/tms/v1/instrumentidentifiers/{instrumentIdentifierTokenId}`

{instrumentIdentifierTokenId} is the instrument identifier token ID returned in the **id** field when you created the instrument identifier token. For more information, see Create an Instrument Identifier (on page 243).

Required Fields for Updating MIT Authorization Options

processingInformation.authorizationOptions.initiator.merchantInitiatedTransaction.previousTransactionId

processingInformation.authorizationOptions.initiator.merchantInitiatedTransaction.originalAuthorizedAmount

processingInformation.authorizationOptions.initiator.merchantInitiatedTransaction.processorTransactionId

Related Information

- API Field Reference for the REST API

REST Example: Updating MIT Authorization Options

Request

```
{
    "processingInformation": {
        "authorizationOptions": {
            "initiator": {
                "merchantInitiatedTransaction": {
                    "previousTransactionId": "123456789012345",
                    "originalAuthorizedAmount": "1",
                    "processorTransactionId": "123456789012345123"
                }
            }
        }
    }
}
```

Response to a Successful Request


```
{
    "_links": {
        "self": {
            "href":
 "https://apitest.cybersource.com/tms/v1/instrumentidentifiers/70100000000162411
11"
        },
        "paymentInstruments": {
            "href":
 "https://apitest.cybersource.com/tms/v1/instrumentidentifiers/70100000000162411
11/paymentinstruments"
        }
    },
    "id": "7010000000016241111",
    "object": "instrumentIdentifier",
    "state": "ACTIVE",
    "card": {
        "number": "411111XXXXXX1111"
    },
    "processingInformation": {
        "authorizationOptions": {
            "initiator": {
                "merchantInitiatedTransaction": {
                    "previousTransactionId": "123456789012345",
                    "originalAuthorizedAmount": "1",
                    "processorTransactionId": "123456789012345123"
                }
            }
        }
    },
    "metadata": {
        "creator": "testrest"
    }
}
```

Network Tokens for Partners

Network tokens are network scheme generated tokens that represent customer card information for secure transactions. Network tokens can be provisioned through Cybersource for partners.

Network tokens that are provisioned by TMS are card-on-file (COF) tokens.

Manage Network Tokens for Partners

This section contains information on managing network tokens for partners.

Network tokens are generated by the network scheme and represent customer card information for secure transactions. Network schemes include but are not limited to Visa, Mastercard, and American Express. Network tokens are mapped to instrument identifier tokens. The minimum card data required to request a network token is the PAN and the expiration date.

You can provision network tokens for a card number or an existing instrument identifier. For example:

- Provision a Network Token for a Card Number (on page 306)
- Provision a Network Token for an Existing Instrument Identifier (on page 309)

You can also retrieve a network tokens for existing payment credentials. For example:

- Retrieve Network Token Payment Credentials (on page 323)

For more information on network tokens, see Network Tokens (on page 23).

Provision a Network Token for a Card Number

This section shows you how to provision a network token for a card number.

Network tokens that are provisioned by TMS are card-on-file (COF) tokens.

Endpoint

Test: `POST https://apitest.cybersource.com/tms/v1/instrumentidentifiers`

Production: `POST https://api.cybersource.com/tms/v1/instrumentidentifiers`

Production in India: `POST https://api.in.cybersource.com/tms/v1/instrumentidentifiers`

Required Fields for Provisioning a Network Token for a Card Number

card.number

card.expirationMonth

card.expirationYear

type

Related Information

- API Field Reference for the REST API

Optional Fields for Provisioning a Network Token for a Card Number

card.securityCode

Related Information

- API Field Reference for the REST API

REST Example: Provisioning a Network Token for a Card Number

Request

```
{
  "type": "enrollable card",
  "card": {
    "number": "4895379987X11515",
    "expirationMonth": "12",
    "expirationYear": "2031",
    "securityCode": "089"
  }
}
```

Response to a Successful Request


```
{
    "_links": {
        "self": {
            "href":
 "https://apitest.cybersource.com/tms/v1/instrumentidentifiers/70300000000149115
15"
        },
        "paymentInstruments": {
            "href":
 "https://apitest.cybersource.com/tms/v1/instrumentidentifiers/70300000000149115
15/paymentinstruments"
        }
    },
    "id": "7030000000014911515",
    "object": "instrumentIdentifier",
    "state": "ACTIVE",
    "tokenizedCard": {
        "state": "ACTIVE",
        "number": "489537XXXXXX5914",
        "expirationMonth": "12",
        "expirationYear": "2022",
        "type": "visa",
        "card": {
            "suffix": "1515",
            "expirationMonth": "12",
            "expirationYear": "2031"
        }
    },
    "card": {
        "number": "489537XXXXXX1515"
    },
    "issuer": {
        "paymentAccountReference": "V0010013019326121174070050420"
    },
    "metadata": {
        "creator": "testrest"
    }
}
```

Provision a Network Token for an Existing Instrument Identifier

This section shows you how to provision a network token for an existing instrument identifier.

Endpoint

Test: `POST https://apitest.cybersource.com/tms/v1/instrumentidentifiers/{instrumentIdentifierTokenId}/enrollment`

Production: `POST https://api.cybersource.com/tms/v1/instrumentidentifiers/{instrumentIdentifierTokenId}/enrollment`

Production in India: `POST https://api.in.cybersource.com/tms/v1/instrumentidentifiers/{instrumentIdentifierTokenId}/enrollment`

`instrumentIdentifierTokenId` is the instrument identifier token ID returned in the **id** field when you created the instrument identifier token. For more information, see Create an Instrument Identifier (on page 243).

Required Fields for Provisioning a Network Token for an Existing Instrument Identifier

card.expirationMonth

card.expirationYear

card.securityCode

instrumentIdentifierTokenId

Include the ID of the instrument identifier token you want to retrieve in the URL path.

type

Related Information

- API Field Reference for the REST API

Optional Fields for Provisioning a Network Token for an Existing Instrument Identifier

card.securityCode

Related information

API Field Reference for the REST API

REST Example: Provisioning a Network Token for an Existing Instrument Identifier

Request

```
{
  "type": "enrollable card",
  "card": {
    "expirationMonth": "12",
    "expirationYear": "2031",
    "securityCode": "089"
  }
}
```

Response to a Successful Request


```
{
  "_links": {
    "self": {
      "href":
 "https://apitest.cybersource.com/tms/v1/instrumentidentifiers/70100000000162411
11"
    },
    "paymentInstruments": {
      "href":
 "https://apitest.cybersource.com/tms/v1/instrumentidentifiers/70100000000162411
11/paymentinstruments"
    }
  },
  "id": "7010000000016241111",
  "object": "instrumentIdentifier",
  "state": "ACTIVE",
  "tokenizedCard": {
    "state": "ACTIVE",
    "type": "visa"
  },
  "card": {
    "number": "411111XXXXXX1111"
  },
  "metadata": {
    "creator": "testrest"
  }
}
```

Provision a Network Token for a Consumer

When you provision a network token for an individual consumer in a wallet, you can manage the network token and payment credentials separately for that consumer. Provisioning network tokens for a consumer is supported for American Express, Mastercard, and Visa. This section describes how to provision a network token for a card number and a consumer ID.

Network tokens that are provisioned by TMS are card-on-file (COF) tokens.

Important: You must be enabled as an ECOM enabler in the Visa Token Service (VTS) to provision a network token with a consumer ID. For more information, contact your Cybersource account representative.

Endpoint

Test: `POST https://apitest.cybersource.com/tms/v2/tokenized-cards`

Production: `POST https://api.cybersource.com/tms/v2/tokenized-cards`

Production in India: `POST https://api.in.cybersource.com/tms/v2/tokenized-cards`

Required Fields for Provisioning a COF Network Token for a Consumer

card.number

card.expirationMonth

card.expirationYear

card.securityCode

createInstrumentIdentifier

Set to `true`.

source

Set to `ONFILE`.

consumerId

When this field is not included, a network token is provisioned only for the PAN in the request.

Related Information

- API Field Reference for the REST API

REST Example: Provisioning a Network Token for a Consumer

Request

```
{
  "createInstrumentIdentifier": true,
  "source": "ONFILE",
  "consumerId": "123456",
  "card": {
    "number": "X895379980000580",
    "expirationMonth": "12",
    "expirationYear": "2023",
    "securityCode": "123"
  }
}
```

Response to a Successful Request


```
{
  "_links": {
    "self": {
      "href": "/tms/v2/tokenized-cards/7030000000014911515"
    },
    "instrumentidentifier": {
      "href": "/tms/v1/instrument-identifier/7030000000042974378"
    }
  },
  "id": "7030000000014911515",
  "object": "tokenizedCard",
  "state": "ACTIVE",
  "source": "ONFILE",
  "enrollmentId": "96eb80a56b76ae1d486e14f40b3d7a01",
  "tokenReferenceId": "059ae2f74835647400c219884b7bc601",
  "paymentAccountReference": "V0010013022298169667504231315",
  "number": "489537XXXXXX9215",
  "expirationMonth": "10",
  "expirationYear": "2031",
  "type": "001",
  "card": {
    "suffix": "0580",
    "expirationMonth": "12",
    "expirationYear": "2023"
  },
  "metadata": {
    "cardArt": {
      "combinedAsset": {
        "id": "d3225702-354a-4f17-8c40-1727de7ffa57",
        "_links": {
          "self": {
            "href":
 "/tms/v2/tokens/7030000000042974378/mdes/assets/card-art-combined"
          }
        }
      }
    },
    "issuer": {
      "name": "METROBANK CARD CORPORATION (A FINANCE COMPANY)",
      "shortDescription": "METROBANK CARD CORPORATION"
    },
    "creator": "testrest"
  }
}
```

Provision a Network Token for a Token

This section shows you how to create a network token for a given token.

Network tokens that are provisioned by TMS are card-on-file (COF) tokens.

Endpoint

Test: `POST https://apitest.cybersource.com/tms/v2/tokenized-cards`

Production: `POST https://api.cybersource.com/tms/v2/tokenized-cards`

Production in India: `POST https://api.in.cybersource.com/tms/v2/tokenized-cards`

Required Fields for Provisioning a Network Token for a Token

card.number

Set to the tokenized card number. When **source** is set to `TOKEN`, this field value must be a digital network token to provision a COF network token.

card.expirationMonth

card.expirationYear

card.securityCode

createInstrumentIdentifier

Set to `true`.

source

Set to `TOKEN`. The value set for **card.number** must be a digital network token to provision a COF network token.

consumerId

When this field is not included, a network token is provisioned only for the PAN in the request.

Related Information

- API Field Reference for the REST API

REST Example: Provisioning a Network Token for a Token

Request

```
{
  "createInstrumentIdentifier": true,
  "source": "TOKEN",
  "card": {
    "number": "X621943123037127",
    "expirationMonth": "12",
    "expirationYear": "2025",
    "securityCode": "123"
  }
}
```

Response to a Successful Request


```
{
  "_links": {
    "self": {
      "href": "/tms/v2/tokenized-cards/7030000000014911515"
    },
    "instrumentidentifier": {
      "href": "/tms/v1/instrument-identifier/7030000000042974378"
    }
  },
  "id": "7030000000014911515",
  "object": "tokenizedCard",
  "state": "ACTIVE",
  "source": "TOKEN",
  "enrollmentId": "96eb80a56b76ae1d486e14f40b3d7a01",
  "tokenReferenceId": "059ae2f74835647400c219884b7bc601",
  "paymentAccountReference": "V0010013022298169667504231315",
  "number": "489537XXXXXX9215",
  "expirationMonth": "10",
  "expirationYear": "2031",
  "type": "001",
  "card": {
    "suffix": "0580",
    "expirationMonth": "12",
    "expirationYear": "2023"
  },
  "metadata": {
    "cardArt": {
      "combinedAsset": {
        "id": "d3225702-354a-4f17-8c40-1727de7ffa57",
        "_links": {
          "self": {
            "href":
 "/tms/v2/tokens/7030000000042974378/mdes/assets/card-art-combined"
          }
        }
      }
    },
    "issuer": {
      "name": "METROBANK CARD CORPORATION (A FINANCE COMPANY)",
      "shortDescription": "METROBANK CARD CORPORATION"
    },
    "creator": "testrest"
  }
}
```

Retrieve a Standalone Network Token

This section contains the required information for partners, merchants, and acquirers to retrieve a standalone network token.

Endpoint

Test: `GET https://apitest.cybersource.com/tms/v2/tokenized-cards/{tokenizedCardId}`

Production: `GET https://api.cybersource.com/tms/v2/tokenized-cards/{tokenizedCardId}`

Production in India: `GET https://api.in.cybersource.com/tms/v2/tokenized-cards/{tokenizedCardId}`

`{tokenizedCardId}` is the tokenized card ID returned in the **id** field when you provisioned the network token. For more information, see Provision a Network Token for a Consumer (on page 312).

REST Example: Retrieving a Standalone Network Token

Request

```
GET https://apitest.cybersource.com/tms/v2/tokenized-cards/223ACDECF1681954E063A2598D0A786D
```

Response to a Successful Request


```
{
    "_links": {
        "self": {
            "href": "/tms/v2/tokenized-cards/223ACDECF1681954E063A2598D0A786D"
        },
        "instrumentIdentifier": {
            "href": "/tms/v1/instrumentidentifiers/7040890000006625091"
        }
    },
    "id": "223ACDECF1681954E063A2598D0A786D",
    "object": "tokenizedCard",
    "state": "ACTIVE",
    "enrollmentId": "FM4MMC00001441368fa429c85a5d4df5ad1875bfd2faa5eb",
    "tokenReferenceId": "DM4MMC1US0000000a7fab5f3a27e49daaf1984f7b49ab2f6",
    "number": "521415XXXXXX5091",
    "expirationMonth": "10",
    "expirationYear": "2027",
    "type": "mastercard",
    "card": {
        "suffix": "0747",
        "expirationMonth": "12",
        "expirationYear": "2025"
    },
    "metadata": {
        "cardArt": {
            "combinedAsset": {
                "id": "9a90ad5f-8577-4a7a-856f-eb66e5437671",
                "_links": {
                    "self": {
                        "href":
 "/tms/v2/tokens/7040890000006625091/mdes/assets/card-art-combined"
                    }
                }
            },
            "brandLogoAsset": {
                "id": "3d7c2517-6b98-4eac-a099-9bd407830e0e",
                "_links": {
                    "self": {
                        "href":
 "/tms/v2/tokens/7040890000006625091/mdes/assets/brand-logo"
                    }
                }
            },
            "issuerLogoAsset": {
                "id": "f607c880-ceaa-4e88-86a7-de854abc8417",
                "_links": {
                    "self": {
```

```
                        "href":
 "/tms/v2/tokens/7040890000006625091/mdes/assets/issuer-logo"
                    }
                }
            },
            "iconAsset": {
                "id": "549a3034-12da-4e85-b0d9-9ad19fec6e2b",
                "_links": {
                    "self": {
                        "href":
 "/tms/v2/tokens/7040890000006625091/mdes/assets/icon"
                    }
                }
            },
            "foregroundColor": "0F0F0F"
        },
        "issuer": {
            "name": "Test IssuerÂ®",
            "shortDescription": "MasterCard Test Bank",
            "longDescription": "Test Bank for MasterCard MTF"
        }
    },
    "source": "ONFILE"
}
```

Delete a Standalone Network Token

This section contains the required information for partners, merchants, and acquirers to delete a standalone network token.

A successful delete response returns an empty `HTTP 204 No Content` status. For information on status codes, see HTTP Status Codes (on page 358).

Important: This is only available for Visa network tokens. You cannot use this feature to delete American Express or Mastercard network tokens.

Endpoint

Test: `DELETE https://apitest.cybersource.com/tms/v2/tokenized-cards/{tokenizedCardId}`

Production: `DELETE https://api.cybersource.com/tms/v2/tokenized-cards/{tokenizedCardId}`

Production in India: `DELETE https://api.in.cybersource.com/tms/v2/tokenized-cards/{tokenizedCardId}`

`{tokenizedCardId}` is the tokenized card ID returned in the **id** field when you provisioned the network token. For more information, see Provision a Network Token for a Consumer (on page 312).

REST Example: Deleting a Standalone Network Token

Request

```
DELETE https://apitest.cybersource.com/tms/v2/tokenized-cards/223ACDECF1681954E063
A2598D0A786D
```

Response to a Successful Request

A successful delete response returns an empty `HTTP 204 No Content` status.

Retrieve Network Token Payment Credentials

This section describes how to retrieve network token payment credentials such as:

- Network token value
- Cryptogram (Visa and Mastercard only)
- Dynamic card verification value (CVV) (American Express only)

Network token payment credentials are returned as a JSON web encryption (JWE) response.

Endpoint

Test: `POST https://apitest.cybersource.com/tms/v2/tokens/{tokenId}/payment-credentials`

Production: `POST https://api.cybersource.com/tms/v2/tokens/{tokenId}/payment-credentials`

Production in India: `POST https://api.in.cybersource.com/tms/v2/tokens/{tokenId}/payment-credentials`

The *{tokenId}* is the token ID returned in the **id** field when you created the customer, payment instrument or instrument identifier token. For more information, see Create an Instrument Identifier (on page 243) for more information.

REST Example: Retrieving Network Token Payment Credentials

Request

```
POST https://apitest.cybersource.com/tms/v2/tokens/7040890000006625091/payment-cre
dentials
```

Response to a Successful Request: BASE 64


```
JraWQiOiI5OWY5YmVjOTlmMzQ1MDJmMDE2NWIyYmJhYWYyODAxNDNhOTI0OWNjIiwiY3R5IjoianNvbiIs
InR5cCI6IkpXVCIsImVuYyI6IkEyNTZHQ00iLCJhbGciOiJSU0EtT0FFUC0yNTYifQ.uYCE2zysWJB8E56
2FGJl4YyotZEHw4Az-2fvhjaUWubuAZ2tmZm44oKUdsfsBLYWInxpMDUsiENTTHG_UJJ25Snhcft6eZGj7
9gW_S55ZAGAi1eYIJA08gr01U7P-1QIzQ5t6dlkTRZElYDiNjypSaVfQPQPODaGNfB04Li7Pt88i-PIspG
afq9P7TgacPyKoIkvM5CwLWbwSZYN_jdFq8hEu4Dy7gqDpf0z-rCdtWggWpFbGwdurDrKCbLBoQ4dY7Ock
Joe2OOWH-O1h_7uZymDDUjnqWFRcHgjxY7bmWJz94i_r4QUaoTQiaaqgyP6A2H3Gmt6Dy4VpIzO2XgLQA.
_cLex9BPstYqqnfe.RMbdjAqWR6HaVZ7USbp6j-KWPC1jGc3Wzk4M_CwJ58X2NNZ5ekUpAvU28_MbqQ2W6
MLhJ7ulgfU5mk9_Y5nvAW6Yh68Ctye2yOhgu_V_33aLmz3iZP5AEGi7HeJVng0hy4EaQHNb92XYXUV1mvF
HJokA4cRaj3eKwh6v-1lRhB4uIgXU62ZanVGGu5c7UkVkf6JiigZarGJiY2DKCRjYnbQYkj4JNFY94JlS5
0wTnGrk3MiAJN9DYIU-6US98zWGJ8VhBwhMuXk1juqVBfifjJMFa_-vnJjGpq1ri2buZ7hMJG-x0PIYoHU
GSFeqNrcLUjJxI0o8lnXfhj7DtfYvNc0e4g5U39xtk-T2TDnQfdekRVxgdxcVR4mZdEqUHBxYUWTSW4Abg
V-fjuCGDCkUoPIgkZ95y4RJhSPZzjZHdulf2Fk3L7e-nto2PB25zUTt_aXeNBSH8zjmaI2ve6D3VN0Scdu
RMl_9PXv1876opHEGqgkKLSTXcTUasXKlzMEiUzLl3p5pN30KnVbryAzuU3hhmIMyyPpEQkp9h3WlD4sc5
oH1E8YtihLlSTtTUNwX5dJuR6iVwpKqFxECqYPtDWlzXQDTedFqdTA4isE3MCs.th9qWPzsevuDYp--06o
POw
```


Response to a Successful Request: Decrypted


```
{
    "_links": {
      "self": {
        "href": "/tms/v2/tokens/7040890000006625091/payment-credentials"
      },
      "tokenized-cards": {
        "href": "/tms/v2/tokenized-cards/223ACDECF1681954E063A2598D0A786D"
      }
    },
    "tokenizedCard": {
      "id": "223ACDECF1681954E063A2598D0A786D",
      "state": "ACTIVE",
      "enrollmentId": "FM4MMC00001441368fa429c85a5d4df5ad1875bfd2faa5eb",
      "tokenReferenceId": "DM4MMC1US0000000a7fab5f3a27e49daaf1984f7b49ab2f6",
      "number": “X214150083525091",
      "expirationMonth": "10",
      "expirationYear": "2027",
      "type": "mastercard",
      "cryptogram": "AJnA9YsVfmS8AAFHiRkBAAADFA==",
      "requestorId": "50162233570",
      "card": {
        "suffix": "0747",
        "expirationMonth": "12",
        "expirationYear": "2025"
      },
      "metadata": {
        "cardArt": {
```

```
        "combinedAsset": {
          "id": "9a90ad5f-8577-4a7a-856f-eb66e5437671",
          "_links": {
            "self": {
              "href":
"/tms/v2/tokens/7040890000006625091/mdes/assets/card-art-combined"
            }
          }
        },
        "brandLogoAsset": {
          "id": "3d7c2517-6b98-4eac-a099-9bd407830e0e",
          "_links": {
            "self": {
              "href":
"/tms/v2/tokens/7040890000006625091/mdes/assets/brand-logo"
            }
          }
        },
        "issuerLogoAsset": {
          "id": "f607c880-ceaa-4e88-86a7-de854abc8417",
          "_links": {
            "self": {
              "href":
"/tms/v2/tokens/7040890000006625091/mdes/assets/issuer-logo"
            }
          }
        },
        "iconAsset": {
          "id": "549a3034-12da-4e85-b0d9-9ad19fec6e2b",
          "_links": {
            "self": {
              "href": "/tms/v2/tokens/7040890000006625091/mdes/assets/icon"
            }
          }
        },
        "foregroundColor": "0F0F0F"
      },
      "issuer": {
        "name": "Test IssuerÂ®",
        "shortDescription": "MasterCard Test Bank",
        "longDescription": "Test Bank for MasterCard MTF"
      }
    },
    "source": "ONFILE"
  },
  "card": {
    "number": "521415XXXXXX5091"
  },
  "issuer": {
```

```
    "paymentAccountReference": "50015018T6JE5ZORJON0QTP9HHMYN"
  },
  "processingInformation": {}
}
```

For more information on decrypting data, see Encrypt and Decrypt Data (on page 357).

Retrieve Network Token AFT Payment Credentials

This section describes how to retrieve the payment credentials for a Visa Token Service (VTS) network token that is used for account funding transactions (AFTs). You can retrieve these payment credentials for a VTS network token:

- VTS network token value
- AFT cryptogram (Visa only)

The VTS network token payment credentials are returned as a JSON Web Encryption (JWE) response.

Important: You must contact your Visa representative to ensure that your system is enabled to retrieve an AFT cryptogram.

Endpoint

Test: `POST https://apitest.cybersource.com/tms/v2/tokens/{tokenId}/payment-credentials`

Production: `POST https://api.cybersource.com/tms/v2/tokens/{tokenId}/payment-credentials`

Production in India: `POST https://api.in.cybersource.com/tms/v2/tokens/{tokenId}/payment-credentials`

The *{tokenId}* is the token ID returned in the **id** field when you create the customer, payment instrument, or instrument identifier token. For more information, see Create an Instrument Identifier (on page 243).

Required Fields for Retrieving Network Token AFT Payment Credentials

paymentCredentialType

Set to `CRYPTOGRAM`.

transactionType

Set to `AFT`.

Related Information

- API Field Reference for the REST API

REST Example: Retrieving Network Token AFT Payment Credentials

Request

```
{
  "paymentCredentialType": "CRYPTOGRAM",
  "transactionType": "AFT"
}
```

Response to a Successful Request: BASE 64

```
eyJraWQiOiJmOTdmOWIyMDZkYzQxNzJmZGRlOWRlNjM5NjMyMWViZTI2NjY4NDc0IiwiY3R5IjoianNvbi
IsInR5cCI6IkpXVCIsImVuYyI6IkEyNTZHQ00iLCJhbGciOiJSU0EtT0FFUC0yNTYifQ.VH168JF7g104i
2dDQ_NAfPEDakHYI54C5mJ-YJItC1VKN3zLlpEZf9VJrUcYrTew6zCNNRKs3treD-Mn3VNK6eYm2YAhkeu
ryYIU2NXZVW3CsZsqeOu5y3c2eO_AiDHRYIKG9MJLoXYqgb-oIotvsmN3p4nYuiYIL21FkdssBV-V66kyM
WPaVZpTqFp0lZBm0NN6p1MoUEybPc5vBpY2I24QZFHJjtX6jxdivA5XmgBgj2DEnue7KlsS3AtMsixmSGO
d46FR0iadGQFQHAvs8lDjfcUj5-vxcFopp8wBOd1sNsItCnqj4WTZFbZGyTe-GnouSCfNX7uC-oHHdxRGd
A.z8TDop4oz778UEUI.yNSpacqPUTGVScMLQJNDjFEvlWdY7YrywJNh7_hCgQBPeW-a1wPnKRsg_9qubr6
_2xYldoNTzrGgiuTj5_cToO5sZF0zFoaDjN9zZMVeppYTB9dfgvQd8AF0AFemhm9lfJVLgjbejPTAHHtJs
pIUGbV4QUPQTAQLLrlT_WSRf-hjQRUHrqoXpkXg-Fjzf1SLNTJb8aFWuvLjNtQMoau-CwSBeJSbCluDuOK
U56TFCZn-9UtE67uDoSgPMgs8XVyAPdSdHLFK8yXzp0mvYqEcR3_p75-MHXsA4qQJuAfXoHjEFsUpqyCkU
BPLE7meGfiuQE9fnlqmtVAtvrdlwdqTVJc_388UJLIGqm0KDuD4gGhdzYuCEnNSrV8VnQ87IggFPMl4kqt
bfCjJ-qvowC6m1a1Wv8rzRo9AqAFXmUpkTaE4vCaPMnRbXuzY0dss81LQlyGri0_eBy6WfOvXe-0yOviDK
FxT_GVeF4nSwkbpxMGGI7aWbVXJ-_r6LBgALtgIbiQZqrWrgLv_FV7DDhY7HnN39R14gve6yxNPu7RCFEV
iy6RUldo_JxPCjNBL6xGXn1ccnxSCQwyvFGvrF1wvm-Ed8vCnImimX4ClV0_ba9lyOUYGIIS7-CWrra7Tv
tISVdYL4J0vPnOHvr8RCwEi5Y6iBtV-Ew6BhHpVd8aFdSTTOfHI7bUtUUE8mFmtnLD1atC-sa2sBjrw4Pl
B-NQMEE4uAGeeccYmYdfPUQdLuMRkzkoPFoNwqnJJ5UC8T6GzEQxZCzfxgTyn_m2toU-Sbb42QKB0BI01c
zad24FuILk93QaVv2SwRX_OQWb8lCxbWFIaxGrHEGLL5YpgqrVKFDLWETD9o-ysQzFweiW-SORrbLhmKs2
xp3GT75d_wxE5IOvLl_A7tJl8I5BQ-aBnTtyZofEJsd6inYHM0TYzj-9YRK4r-grA3xvytI50dwkJJJ4Ju
```

```
9wJGuxAdKvbxYrkwKgn0HKBE58BGjKElT6j0gLXU0bsuDPo1E_OSj9NGyUeB9LKGIptGNQOBeNi06pYwTG
MkjfPZi9S1Zv9wnr7CwcSe_xImu4f192co8hJKUIXosbAZturQ-8tR5BOZSKrLgO4vCQfJmYxYC1IVaDWk
nhTe78ev9qVZD0HP7H_gqXPHY5O5DfBr-pxFCnYt-oQwt24hc2_r13JvzekYP3bjoUV9YsoO8-eJ1A4NyO
8Bk2qUZMEWca_szCz_uCiozwdXWc0HVIUWsn1BvrbRKzM91UxrEZkKmP3xaOPQv91FMTgVvJEG7pxrObH4
9gGgNO8rU-J46Hd01YcAFzvgqYTSxGaCF5lgmgbXzPK16AWfGYKPXEUExZERvVhIxhQRKpBKD3KJFO772b
lKdfTjPl_QqnUJ5_wy7QScUP_7okJZduC9JgNy-S4htThQHL4AW2XO6RzInaup4fZ3tkaIdNTacQQwFcd-
kJ3Bj049VcV2BkMR1oP2__P2siMQ8H3oD8IZAb9S0buFezp1KVNflKNW2fTQDXEESzFZi-aL4OxMSll6yu
jkXtKtjOZoMYZawNeuLGtltZQZ8BkhvzRTrwuBHWNcbhCN9-0qwC4YuBZ8FD1Rpd1vPNOPMpEMblq9BKf_
wlGeXUMK0XzsVdQLfcBd6aDHAysFnt8Y961oYTX8kA.uT1yJ32LL_T1TifxlH3HdA
```


Response to a Successful Request: Decrypted


```
{
  "_links": {
    "self": {
      "href":
 "/tms/v2/tokens/1CD10F58FB7C7DB8E063A2598D0A1405/payment-credentials"
    },
    "tokenized-cards": {
      "href": "/tms/v2/tokenized-cards/1CD10F58FB837DB8E063A2598D0A1405"
    }
  },
  "tokenizedCard": {
    "id": "1CD10F58FB837DB8E063A2598D0A1405",
    "state": "ACTIVE",
    "enrollmentId": "da1fb810b1b3e01db5b215de5261df01",
    "tokenReferenceId": "090673c4811a91960f021ad3a24e2e01",
    "number": "X895370017256311",
    "expirationMonth": "12",
    "expirationYear": "2031",
    "type": "visa",
    "cryptogram": "AwAAAADi9THGeHw4ARJ4QAoAAAA=",
    "eci": "07",
    "requestorId": "40010052236",
    "card": {
      "suffix": "7161",
      "expirationMonth": "12",
      "expirationYear": "2030"
    },
    "metadata": {
      "cardArt": {
        "combinedAsset": {
          "id": "8f64614def1a41d39ea8acae4616bf6f",
          "_links": {
            "self": {
              "href":
 "/tms/v2/tokens/1CD10F58FB7C7DB8E063A2598D0A1405/vts/assets/card-art-combined"
```

```
            }
          }
        },
        "brandLogoAsset": {
          "id": "00000000000000000000000000001071",
          "_links": {
            "self": {
              "href":
 "/tms/v2/tokens/1CD10F58FB7C7DB8E063A2598D0A1405/vts/assets/brand-logo"
            }
          }
        },
        "foregroundColor": "1af0f0"
      },
      "issuer": {
        "shortDescription": "shortDescription",
        "longDescription": "longDescription"
      }
    },
    "source": "ONFILE"
  },
  "card": {
    "number": "X622943123037161"
  },
  "issuer": {
    "paymentAccountReference": "V0010013024026372674590402581"
  },
  "processingInformation": {
    "commerceIndicator": "internet"
  }
}
```


For more information on decrypting data, see Encrypt and Decrypt Data (on page 357).

Retrieve an Instrument Identifier

This section shows you how to retrieve an instrument identifier.

Endpoint

Test: `GET https://apitest.cybersource.com/tms/v1/instrumentidentifiers/{instrumentIdentifierTokenId}`

Production: `GET https://api.cybersource.com/tms/v1/instrumentidentifiers/{instrumentIdentifierTokenId}`

Production in India: `GET https://api.in.cybersource.com/tms/v1/instrumentidentifiers/{instrumentIdentifierTokenId}`

{instrumentIdentifierTokenId} is the instrument identifier token ID returned in the **id** field when you created the instrument identifier token. For more information, see Create an Instrument Identifier (on page 243).

REST Example: Retrieving an Instrument Identifier

Request

```
GET https://apitest.cybersource.com/tms/v1/instrumentidentifiers/7030000000046226
486
```

Response to a Successful Request


```
{
    "_links": {
        "self": {
            "href":
 https://apitest.visaacceptance.com/tms/v1/instrumentidentifiers/7030000000046226
486
        },
        "paymentInstruments": {
            "href":
 https://apitest.visaacceptance.com/tms/v1/instrumentidentifiers/70300000000462264
86/paymentinstruments
        }
    },
    "id": "7030000000046226486",
    "object": "instrumentIdentifier",
    "state": "ACTIVE",
    "tokenizedCard": {
        "state": "ACTIVE",
        "enrollmentId": "80efd04a838265dcdf9d114eb542be01",
        "tokenReferenceId": "003769ea947eb057c13c192f1d26f002",
        "number": "489537XXXXXX2655",
        "expirationMonth": "12",
        "expirationYear": "2030",
        "type": "visa",
        "card": {
            "suffix": "6486",
            "expirationMonth": "12",
            "expirationYear": "2030"
        }
    },
    "card": {
        "number": "489537XXXXXX6486"
    },
    "issuer": {
        "paymentAccountReference": "V0010013020037567013722096631"
    },
    "metadata": {
        "creator": "testacct"
    }
}
```

Delete an Instrument Identifier

This section shows you how to delete an instrument identifier.

Endpoint

Test: `DELETE https://apitest.cybersource.com/tms/v1/instrumentidentifiers/{instrumentIdentifierTokenId}`

Production: `DELETE https://api.cybersource.com/tms/v1/instrumentidentifiers/{instrumentIdentifierTokenId}`

Production in India: `DELETE https://api.in.cybersource.com/tms/v1/instrumentidentifiers/{instrumentIdentifierTokenId}`

`{instrumentIdentifierTokenId}` is the instrument identifier token ID returned in the **id** field in the **id** field when you created the instrument identifier token. For more information, see Create an Instrument Identifier (on page 243).

REST Example: Deleting an Instrument Identifier

Request

```
DELETE https://apitest.cybersource.com/tms/v1/instrumentidentifiers/7010000000016241111
```

Response to a Successful Request

A successful delete response returns an empty `HTTP 204 No Content` status. For more information, see HTTP Status Codes (on page 358).

Network Token Provision Failures

Lost and Stolen Card Response

Network Token Provision Failure Reason Codes and Descriptions

Reason Code	Description
INVALID_REQUEST	The network token provision request contained invalid data.
CARD_VERIFICATION_FAILED	The network token provision request contained data that could not be verified.
CARD_NOT_ELIGIBLE	Card cannot be used currently with issuer for tokenization.
CARD_NOT_ALLOWED	Card cannot be used currently with card association for tokenization.
DECLINED	Card cannot be used currently with issuer for tokenization.
SERVICE_UNAVAILABLE	The network token service was unavailable or timed out.
SYSTEM_ERROR	An unexpected error occurred with network token service, check configuration.


```
{
    "_links": {
        "self": {

  "href": "https://
apitest.cybersource.com/tms/v1/instrumentidentifiers/7030000000041554452"
        },
        "paymentInstruments": {

  "href": "https://
apitest.cybersource.com/tms/v1/instrumentidentifiers/7030000000041554452/paymentin
struments"
        }
    },
    "id": "7030000000041554452",
    "object": "instrumentIdentifier",
    "state": "ACTIVE",
    "tokenizedCard": {
        "state": "UNPROVISIONED",
        "reason": "CARD_NOT_ELIGIBLE",
        "type": "visa"
    },
    "card": {
        "number": "400555XXXXXX4452"
    },
    "metadata": {
        "creator": "testrest"
    }
}
```

Issuer Decline Response


```
{
    "_links": {
        "self": {

  "href": "https://
apitest.cybersource.com/tms/v1/instrumentidentifiers/7030000000051790079"
        },
        "paymentInstruments": {

  "href": "https://
apitest.cybersource.com/tms/v1/instrumentidentifiers/7030000000051790079/paymentin
struments"
        }
    },
    "id": "7030000000051790079",
    "object": "instrumentIdentifier",
    "state": "ACTIVE",
    "tokenizedCard": {
        "state": "UNPROVISIONED",
        "reason": "CARD_NOT_ALLOWED",
        "type": "visa"
    },
    "card": {
        "number": "462294XXXXXX0079"
    },
    "metadata": {
        "creator": "testrest"
    }
}
```

Past Expiration Date Response


```
{
    "_links": {
        "self": {

  "href": "https://
apitest.cybersource.com/tms/v1/instrumentidentifiers/7030000000224170019"
        },
        "paymentInstruments": {

  "href": "https://
apitest.cybersource.com/tms/v1/instrumentidentifiers/7030000000224170019/paymentin
struments"
        }
    },
    "id": "7030000000224170019",
    "object": "instrumentIdentifier",
    "state": "ACTIVE",
    "tokenizedCard": {
        "state": "UNPROVISIONED",
        "reason": "CARD_NOT_ALLOWED",
        "type": "visa"
    },
    "card": {
        "number": "476134XXXXXX0019"
    },
    "metadata": {
        "creator": "testrest"
    }
}
```

Issuer Not Participating Response


```
{
    "_links": {
        "self": {

  "href": "https://
apitest.cybersource.com/tms/v1/instrumentidentifiers/7030000000224170019"
        },
        "paymentInstruments": {

  "href": "https://
apitest.cybersource.com/tms/v1/instrumentidentifiers/7030000000224170019/paymentin
struments"
        }
    },
    "id": "7030000000224170019",
    "object": "instrumentIdentifier",
    "state": "ACTIVE",
    "tokenizedCard": {
        "state": "UNPROVISIONED",
        "reason": "CARD_NOT_ALLOWED",
        "type": "visa"
    },
    "card": {
        "number": "476134XXXXXX0019"
    },
    "metadata": {
        "creator": "testrest"
    }
}
```

Push Provisioning for Network Tokens

Important: This feature is in pilot phase. You have early access to this feature even though it might contain bugs or unfinished work. Please consider the risk when using this feature.

Push provisioning connects you with participating banks to enable the secure transfer of customer and payment information that is stored by banks. Using push provisioning, the issuer can provide credentials straight to your customer in seconds.

Prerequisites

Before using the push provisioning service, you must meet these requirements:

- You must be configured for TMS. See Token Management Service Onboarding (on page 60).
- Network tokens must be enabled. For more information, see Network Token Enablement (on page 69).
- Push provisioning must be enabled with the card brand.
- The issuer must be integrated with the card brand.

Provision a Network Token with Push Provisioning

This section shows you how to provision a network token with push provisioning.

Endpoint

Test: `POST https://apitest.cybersource.com/tms/v2/tokenized-cards`

Production: `POST https://api.cybersource.com/tms/v2/tokenized-cards`

Production in India: `POST https://api.in.cybersource.com/tms/v2/tokenized-cards`

Use the Push Provisioning Instrument Identifier in Authorizations

You can include the instrument identifier that is returned when you create or retrieve a network tokenized card with push provisioning in an authorization. For more information, see Authorize a Payment with an Instrument Identifier (on page 277).

You can also create other token types, such as customer, shipping address, and payment instrument tokens, when you send the authorization request. For more information, see REST Example: Authorizing a Payment with an Instrument Identifier While Creating TMS Tokens (on page 280)

Required Fields for Provisioning a Network Token with Push Provisioning

accountReferenceId

card.type

Set to `001`.

createInstrumentIdentifier

Set to `true`.

source

Set to `ISSUER`.

Related Information

- API Field Reference for the REST API

Optional Fields for Provisioning a Network Token with Push Provisioning

passcode.value

Related Information

- API Field Reference for the REST API

REST Example: Provisioning a Network Token with Push Provisioning

Request

```
{
  "accountReferenceId": "70369945856381846000 1",
  "createInstrumentIdentifier": true,
  "source": "ISSUER",
  "card": {
    "type": "001"
  }
}
```

Request with Passcode

```
{
  "source": "ISSUER",
  "accountReferenceId": "70369945856381846000 1",
  "card": {
    "type": "001"
  },
  "passcode": {
    "value": "123456"
  },
  "createInstrumentIdentifier": true
}
```

Response to a Successful Request


```
{
    "_links": {
        "self": {
            "href":
 "https://apitest.cybersource.com/tms/v2/tokenized-cards/139C09B1970689FAE0633F36C
F0A2D7B"
        },
        "instrumentIdentifier": {
            "href":
 "https://apitest.cybersource.com/tms/v1/instrumentidentifiers/70100000000162411
11"
        }
    },
    "id": "139C09B1970689FAE0633F36CF0A2D7B",
    "object": "tokenizedCard",
    "state": "ACTIVE",
    "enrollmentId": "ja9mejoqszrqfubwy9mqz4ot4fnlvgpp",
    "tokenReferenceId": "uvfofwjor4nobycjf5cy9cwfyzu5pipa",
    "number": "404626XXXXXX0572",
    "expirationMonth": "03",
    "expirationYear": "2025",
    "type": "visa",
    "card": {
        "suffix": "4608",
        "expirationMonth": "03",
        "expirationYear": "2025"
    },
    "source": "ISSUER",
    "accountReferenceId": "703699458563818460001"
}
```

Card Art

Important: This feature is in pilot phase. You have early access to this feature even though it might contain bugs or unfinished work. Please consider the risk when using this feature.

You can choose to display card art provided by TMS to help your customers identify the card that they are selecting. Cybersource recommends that card art be shown in all cardholder-facing interactions where it applies.

Card art is available for these card types:

- American Express
- Mastercard
- Visa

Retrieve Card Art

This section describes how to retrieve card assets.

Endpoint

Test: `POST https://apitest.cybersource.com/tms/v2/tokens/{tokenId}/{provider}/assets/{asset.type}`

Production: `POST https://api.cybersource.com/tms/v2/tokens/{tokenId}/{provider}/assets/{asset.type}`

Production in India: `POST https://api.in.cybersource.com/tms/v2/tokens/{tokenId}/{provider}/assets/{asset.type}`

The *{tokenId}* is the instrument identifier ID returned in the **id** field when you created the TMS token.

The *{provider}* is the provider of the card for which you want to retrieve card art. Possible values:

- `aets`: American Express
- `mdes`: Mastercard
- `mscof`: Mastercard
- `vts`: Visa

The *`{asset.types}`* is the card art asset that you retrieve. Possible values:

- `card-art-combined`: background image, brand logo, and issuer logo
- `card-background`: background image
- `card-issuer-logo`: issuer logo
- `card-brand-logo`: brand logo
- `card-co-brand-logo`: co-branded card logo
- `card-icon`: card brand icon

The availability of card asset types depends on the provider:

Card Art Assets and Providers

Card Art Asset	aets	mdes	mscof	vts
`card-art-combined`	✓	✓	✓	✓
`card-background`	✗	✓	✗	✓
`card-issuer-logo`	✓	✓	✗	✗
`card-brand-logo`	✗	✓	✗	✓
`card-co-brand-logo`	✗	✓	✗	✗
`card-icon`	✗	✓	✗	✗

REST Example: Retrieving Card Art Assets

Request for the Issuer Logo

```
GET https://apitest.cybersource.com/tms/v2/tokens/{tokenId}/{provider}/assets/card-issuer-logo
```

Response to a Successful Request

```
{
    "id": "3883d6a112284123b8b23ec595670eb7",
    "type": "issuerLogo",
    "provider": "vts",
    "content": [
        {
            "type": "image/png",
            "data": "R0l...aP=",         //Base-64 encoded data
            "width": 200,   // Include if provided by the issuer
            "height": 200   // Include if provided by the issuer
        }
    ]
}
```

BIN Lookup Service and TMS

When some types of tokens are provisioned, TMS returns the BIN details that are provided by the BIN Lookup Service. This section shows you how to retrieve the BIN information provided by the BIN Lookup Service for a PAN or network token.

Important: You must be enabled for network tokenization to retrieve BIN details using TMS.

TMS returns BIN data when you send a request to create or retrieve these token types:

- Payment instrument tokens (on page 206)
- Instrument identifier tokens (on page 242)

For more information about using the BIN Lookup Service, see the *BIN Lookup Service Developer Guide*

Endpoints

Instrument Identifier Tokens

Test: `GET https://apitest.cybersource.com/tms/v1/instrumentidentifiers/{instrumentIdentifierTokenId}?retrieveBinDetails=true`

Test: `POST https://apitest.cybersource.com/tms/v1/instrumentidentifiers?retrieveBinDetails=true`

Production: `GET https://api.cybersource.com/tms/v1/instrumentidentifiers/{instrumentIdentifierTokenId}?retrieveBinDetails=true`

Production: `POST https://api.cybersource.com/tms/v1/instrumentidentifiers?retrieveBinDetails=true`

Production in India: `GET https://api.in.cybersource.com/tms/v1/instrumentidentifiers/{instrumentIdentifierTokenId}?retrieveBinDetails=true`

Production in India: `POST https://api.in.cybersource.com/tms/v1/instrumentidentifiers?retrieveBinDetails=true`

Payment Instrument Tokens

Test: `GET https://apitest.cybersource.com/tms/v1/paymentinstruments/{paymentInstrumentTokenId}?retrieveBinDetails=true`

Test: `POST https://apitest.cybersource.com/tms/v1/paymentinstruments?retrieveBinDetails=true`

Production: `GET https://api.cybersource.com/tms/v1/paymentinstruments/{paymentInstrumentTokenId}?retrieveBinDetails=true`

Production: `POST https://api.cybersource.com/tms/v1/paymentinstruments?retrieveBinDetails=true`

Production in India: `GET https://api.in.cybersource.com/tms/v1/paymentinstruments/{paymentInstrumentTokenId}?retrieveBinDetails=true`

Production in India: `POST https://api.in.cybersource.com/tms/v1/paymentinstruments?retrieveBinDetails=true`

`{instrumentIdentifierTokenId}` and *`{paymentInstrumentTokenId}`* are the token IDs that are returned in the **id** field when you created the token.

REST Example: Retrieving an Instrument Identifier with BIN Details

Request

```
GET https://apitest.cybersource.com/tms/v1/instrumentidentifiers/7049989999918257179?retrieveBinDetails=true
```

Response to a Successful Request


```
{
    "_links": {
        "self": {
            "href": "/tms/v1/instrumentidentifiers/7049989999918257179"
        },
        "paymentInstruments": {
            "href":
 "/tms/v1/instrumentidentifiers/7049989999918257179/paymentinstruments"
        },
        "tokenized-cards": {
            "href": "/tms/v2/tokenized-cards/23589328AA5A15CDE063A2598D0A702B"
        }
    },
    "id": "7049989999918257179",
    "object": "instrumentIdentifier",
    "state": "ACTIVE",
    "tokenizedCard": {
        "id": "23589328AA5A15CDE063A2598D0A702B",
        "state": "ACTIVE",
        "enrollmentId": "abea32809655e12383cd1b006e119d01",
        "tokenReferenceId": "131db9889f4c503b240519cca8f35901",
        "number": "489537XXXXXX5398",
        "expirationMonth": "12",
        "expirationYear": "2030",
        "type": "visa",
        "card": {
            "suffix": "7179",
            "expirationMonth": "12",
            "expirationYear": "2030"
        },
        "metadata": {
            "cardArt": {
                "combinedAsset": {
                    "id": "8f64614def1a41d39ea8acae4616bf6f",
                    "_links": {
                        "self": {
                            "href":
 "/tms/v2/tokens/7049989999918257179/vts/assets/card-art-combined"
                        }
                    }
                },
                "brandLogoAsset": {
                    "id": "00000000000000000000000000001071",
                    "_links": {
                        "self": {
                            "href":
 "/tms/v2/tokens/7049989999918257179/vts/assets/brand-logo"
```

```
                        }
                    }
                },
                "foregroundColor": "1af0f0"
            },
            "issuer": {
                "shortDescription": "shortDescription",
                "longDescription": "longDescription"
            }
        },
        "source": "ONFILE"
    },
    "card": {
        "number": "462294XXXXXX7179"
    },
    "issuer": {
        "paymentAccountReference": "V0010013024023377525412642508"
    },
    "metadata": {
        "creator": "mid"
    },
    "_embedded": {
        "binLookup": {
            "issuer": {
                "country": "US"
            },
            "status": "COMPLETED",
            "paymentAccountInformation": {
                "card": {
                    "type": "001",
                    "brandName": "VISA",
                    "credentialType": "PAN",
                    "cardType": "VISA"
                }
            }
        }
    }
}
```

Using Token Management Service with Wallet Apps

Use the TMS API features to create an e-wallet app for your customers.

Manage Tokens with Wallet Apps

This section contains information for on managing tokens using wallet apps.

Use the Token Management Service (TMS) API features to create an e-wallet app for your customers. You can use your e-wallet to create, update, patch, and delete payment methods. Use the TMS API to:

Create a New Customer Account

Use the TMS API in your e-wallet app when creating a new customer account to store customer information securely.

1. Call the `POST /tms/v2/customers` endpoint to create a new customer in the e-wallet app customer sign-up flow.
2. The request returns the customer token. Store the customer token with the customer profile information in your database.

Add a New Shipping Address

Use the TMS API in your e-wallet app to store a customer's new shipping address.

1. When you collect the new customer's shipping address, use the customer token from the Create a New Customer Account (on page 352) step to create a shipping address for that customer.

2. Call `POST /tms/v2/customers/{customerTokenId}/shipping-addresses`.

Edit or Delete a Shipping Address

Use the TMS API in your e-wallet app to edit or delete a customer's shipping address.

1. To get all addresses, call: `GET /tms/v2/customers/{customerTokenId}/shipping-address.` The first record is the default.

2. To add an address, call: `POST /tms/v2/customers/{customerTokenId}/shipping-addresses`. This adds a non-default shipping address. If it is the customer's first address, it becomes the default address.

3. To edit an address, call: `PATCH /tms/v2/customers/{customerTokenId}/shipping-addresses/{shippingAddressTokenId}`.

4. To delete an address, call: `DELETE /tms/v2/customers/{customerTokenId}/shipping-addresses/{shippingAddressTokenId}.`

5. To set an address as the default address, call: `PATCH /tms/v2/customers/{customerTokenId}/shipping-addresses/{shippingAddressTokenId}` and set the value of the request field `default` to `true`.

Create a New Payment Instrument with the Payments API

Use the payments API in your e-wallet app to store the customer's payment method information in a payment instrument.

1. Use the payment instrument you created in the call from the Add a Default Payment Instrument with Validated Payment (on page 161) step to create a new payment method.

2. Call: `POST /pts/v2/payments` and pass the instrument identifier token, card type, and expiration date in the request.
   If this is the first payment method, it becomes the customer's default.

3. Store the card expiration date and last 4 digits with the customer profile information in your database.

Edit or Delete a Payment Method

Use the TMS API in your e-wallet app to retrieve a customer's payment method and allow the customer to delete or edit the payment method.

1. To retrieve the customer's default payment method, call: `GET /tms/v2/customers/{customerTokenId}/payment-instruments`.
   The first record is the default payment method. The remaining payment methods are the non-default payment methods.

2. To delete a payment method, call: `DELETE /tms/v2/customers/{customerTokenId}/payment-instruments/{paymentInstrumentTokenId}`

3. To edit a payment method, call: `PATCH /tms/v2/customers/{customerTokenId}/payment-instruments/{paymentInstrumentTokenId}`

4. Include the updated payment method details in the call.

Change the Default Payment Method

Use the TMS API in your e-wallet app to change the customer's default payment method.

1. To get all payment methods, call: `GET /tms/v2/customers/{customerTokenId}/payment-instruments`.
   The first result is the default payment instrument. The remaining payment methods are the non-default payment methods.

2. To make a non-default payment method the default, call: `PATCH /tms/v2/customers/{customerTokenId}/payment-instruments/{paymentInstrumentTokenId}` and set the value of the request field `default` to `true`.

Add a New Payment Method Address

Use the TMS API in your e-wallet app to list the customer's addresses or add a new address for a payment method.

1. To list the customer's existing billing addresses, call: `GET /tms/v2/customers/{customerTokenId}/payment-instruments`.
   The call returns all of the payment methods for a customer, including their billing address details.

2. To list the customer's existing shipping addresses, call: `GET /tms/v2/customers/{customerTokenId}/payment-instruments`.
   This call returns all the shipping address details for a customer.

3. To add a new address, call: `POST /tms/v2/customers/{customerTokenId}/shipping-addresses`.

4. To add an address to the payment method created in Create a New Payment Instrument with the Payments API (on page 353), call: `PATCH /tms/v2/customers/{customerTokenId}/payment-instruments/{paymentInstrumentTokenId}`.

5. Pass the ID of the instrument identifier created in Create a New Payment Instrument with the Payments API (on page 353) and the card expiration date.

View Wallet

Use the TMS API in your e-wallet app to view wallet.

1. Call `GET /tms/v2/customers/{customerTokenId}/payment-instruments`.
   The first record is the default payment instrument.

2. Retrieve the last four digits of the card number from your database or call the payment identifier endpoint.

Payments with Tokens and Wallet Apps

This section contains information for on making payments with tokens using wallet apps.

Use the TMS API features to create an e-wallet app for your customers. You can use your e-wallet to authorize a payment. For example:

- Authorize a Payment (on page 356)

Authorize a Payment

Use the TMS API in your e-wallet app to authorize a payment.

1. To get the customer’s default shipping address, call: `GET /tms/v2/customers/{customerTokenId}/shipping-address`.
   The first record is the default.

2. To retrieve the customer's default payment method, call: `GET /tms/v2/customers/{customerTokenId}/payment-instruments`.
   The first record is the default.

3. Finally, when the customer clicks the **Place Order** button, call: `POST /pts/v2/payments` pass the customer token, payment instrument token, and shipping address token.

Reference Information

Encrypt and Decrypt Data

1. Send a POST request to token cryptogram resource at `/tms/v2/tokens/instrumentIdentifierTokenId/payment-credentials`. The response is BASE 64 encoded text. For example:

```
JraWQiOiI5OWY5YmVjOTlmMzQ1MDJmMDE2NWIyYmJhYWYyODAxNDNhOTI0OWNjIiwiY3R5IjoianN
vbiIsInR5cCI6IkpXVCIsImVuYyI6IkEyNTZHQ00iLCJhbGciOiJSU0EtT0FFUC0yNTYifQ.uYCE2
zysWJB8E562FGJl4YyotZEHw4Az-2fvhjaUWubuAZ2tmZm44oKUdsfsBLYWInxpMDUsiENTTHG_UJ
J25Snhcft6eZGj79gW_S55ZAGAi1eYIJA08gr01U7P-1QIzQ5t6dlkTRZElYDiNjypSaVfQPQPODa
GNfB04Li7Pt88i-PIspGafq9P7TgacPyKoIkvM5CwLWbwSZYN_jdFq8hEu4Dy7gqDpf0z-rCdtWgg
WpFbGwdurDrKCbLBoQ4dY7OckJoe2OOWH-O1h_7uZymDDUjnqWFRcHgjxY7bmWJz94i_r4QUaoTQi
aaqgyP6A2H3Gmt6Dy4VpIzO2XgLQA._cLex9BPstYqqnfe.RMbdjAqWR6HaVZ7USbp6j-KWPC1jGc
3Wzk4M_CwJ58X2NNZ5ekUpAvU28_MbqQ2W6MLhJ7ulgfU5mk9_Y5nvAW6Yh68Ctye2yOhgu_V_33a
Lmz3iZP5AEGi7HeJVng0hy4EaQHNb92XYXUV1mvFHJokA4cRaj3eKwh6v-1lRhB4uIgXU62ZanVGG
u5c7UkVkf6JiigZarGJiY2DKCRjYnbQYkj4JNFY94JlS50wTnGrk3MiAJN9DYIU-6US98zWGJ8VhB
whMuXk1juqVBfifjJMFa_-vnJjGpq1ri2buZ7hMJG-x0PIYoHUGSFeqNrcLUjJxI0o8lnXfhj7Dtf
YvNc0e4g5U39xtk-T2TDnQfdekRVxgdxcVR4mZdEqUHBxYUWTSW4AbgV-fjuCGDCkUoPIgkZ95y4R
JhSPZzjZHdulf2Fk3L7e-nto2PB25zUTt_aXeNBSH8zjmaI2ve6D3VN0ScduRMl_9PXv1876opHEG
qgkKLSTXcTUasXKlzMEiUzLl3p5pN30KnVbryAzuU3hhmIMyyPpEQkp9h3WlD4sc5oH1E8YtihLlS
TtTUNwX5dJuR6iVwpKqFxECqYPtDWlzXQDTedFqdTA4isE3MCs.th9qWPzsevuDYp--06oPOw
```

2. Decode the BASE 64 encoded response. The response is a decoded JWE response with an encrypted payload. For example:

```
{
  "kid": "99f9bec99f34502f0165b2bbaaf280143a9249",
  "cty": "json",
  "typ": "JWT",
  "enc": "A256GCM",
  "alg": "RSA-OAEP-256"
}
<Encrypted payload>
```

3. Decrypt the JWE encrypted payload. The response is the decrypted payload. For example:

```
{
  "_links": {
    "self": {
```

```
  "href": "/tms/v2/tokens/A560EECDED74936DE0533F36CF0ACEBC/payment-credentia
ls"
    }
  },
  "tokenizedCard": {
    "state": "ACTIVE",
    "number": "4X24XX7118382281",
    "expirationMonth": "11",
    "expirationYear": "2022",
    "type": "visa",
    "cryptogram": "AF1ajnoLKKj8AAKhssPUGgADFA\u003d\u003d",
    "requestorId": "ABCD",
    "card": {
      "suffix": "2382",
      "expirationMonth": "12",
      "expirationYear": "2018"
    }
  },
  "card": {
    "number": "402400XXXXXX2382"
  },
  "issuer": {
    "paymentAccountReference": "V0000000000005109162731718000"
  }
}
```


HTTP Status Codes

A request response returns one of the following HTTP status codes:

- `200`: The standard response for a successful HTTP request. In a `GET` request, the response will contain an empty entity corresponding to the requested resource. In a `POST` request, the response will contain an entity describing or containing the result of the action.
- `201`: The request was fulfilled and resulted in a new resource being created. If you get this HTTP status code for an unsuccessful transaction, Cybersource or the merchant's processor probably marked this transaction as under review, declined, or failed.
- `204`: The server fulfilled the request but does not need to return a body.
- `403`: Forbidden Response: The profile might not have permission to perform the operation.
- `404`: Token Not Found. The token ID may not exist or was entered incorrectly.
- `409`: Conflict. The token is linked to a Payment Instrument.

- `410`: Token not available The token has been deleted.
- `424`: Failed Dependency: The profile represented by the profile ID may not exist or the profile ID was entered incorrectly.
- `500`: Unexpected error.
- `502`: Bad gateway. There was a token deletion error from the Visa Token Service (VTS).

Supported Processors

The processors listed below support customer and instrument identifier tokens, unless noted otherwise.

Processor	Payment Method
Visa Platform Connect	• Credit card—supports 0.00 pre-authorizations for Visa and Mastercard cards. • Credit card—supports 1.00 pre-authorizations for American Express, Discover, Diners Club, and JCB card types. • Debit card and prepaid card. • Payouts.