Payments

REST API
Visa Platform Connect

cybersource
A Visa Solution

Developer Guide

Cybersource Contact Information

For general information about our company, products, and services, go to https://www.cybersource.com.

For sales questions about any Cybersource service, email sales@cybersource.com or call 650-432-7350 or 888-330-2300 (toll free in the United States).

For support information about any Cybersource service, visit the Support Center: https://www.cybersource.com/support

Copyright

Restricted Rights Legends

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

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 Unpublished rights reserved under the copyright laws of the United States.

Trademarks

Confidentiality Notice

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

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

Revision

Version: 24.04

Contents

Recent Revisions to This Document

24.04

Relaxed Requirements for Address Data and Expiration Date	**Added a new section *Relaxed Requirements for Address Data and Expiration Date in Payment Transactions* on page 44.**

24.03

Foreign Retail Indicator	**Added new authorization and capture use cases based on the foreign merchant mandate by Visa. See *Authorizations with Foreign Merchants* on page 68 and *Captures with Foreign Merchants* on page 95.**
Timeout Authorization Reversals	**Added an important note about the wait time before requesting a timeout authorization reversal. See *Timeout Authorization Reversals* on page 91.** **Added an important note about the wait time before requesting a timeout authorization reversal. See *Timeout Authorization Reversals* on page 91.**
Token-Based Processing	**Added support for authorizing a payment ignoring a network token. See *Authorize a Payment While Ignoring Network Token* on page 439.**

24.02

Refunds	**The time limit for refunds has been updated. See these topics:** • *Credits* on page 28 in the section on refunds • *Timeout Authorization Reversals* on page 91 • *Refunds* on page 102 • *Timeout Voids for a Capture, Sale, Refund, or Credit* on page 111

24.01

Processing Payments Using Credentials	**Updated the Mastercard Standing Order Payments and the Mastercard Subscription Payments to include TMS information. See *Mastercard Standing Order Payments* on page 386 and *Mastercard Subscription Payments* on page 393.**
Timeout Reversals	**Added timeout authorization reversals. See *Timeout Authorization Reversals* on page 91.**
Timeout Voids	**Added timeout voids. See *Timeout Voids for a Capture, Sale, Refund, or Credit* on page 111.**

23.10

This revision contains only editorial changes and no technical updates.

23.09

Authorizations with Strong Customer Authentication Exemption	**Added support for *Authorizations with Strong Customer Authentication Exemption* on page 73.**

23.08

Added Credentialed Transactions TMS Tokens	**Added information about processing merchant-initiated transactions with TMS Tokens.** • *Storing Customer Credentials with a CIT and TMS* on page 278

- *Merchant-Initiated Delayed Transaction with TMS* on page 289
- *Merchant-Initiated Incremental Transaction with TMS* on page 301
- *Merchant-Initiated No-Show Transaction with TMS* on page 314
- *Merchant-Initiated Reauthorization Transactions with TMS* on page 327
- *Merchant-Initiated Resubmission Transaction with TMS* on page 339
- *Merchant-Initiated Installment Payment with TMS* on page 360
- *Merchant-Initiated Recurring Payments with TMS* on page 378
- *Merchant-Initiated Unscheduled COF Payments with TMS* on page 412

23.07

Added notes about merchants in India not being permitted to store PANs. For details, see *Token Management Service* and *Authorizations with Payment Network Tokens* on page 60.
23.07.01: Fixed broken links.

23.06

This revision contains only editorial changes and no technical updates.

23.05

This revision contains only editorial changes and no technical updates.

23.04

Authorization Reversals	**Updated *Authorization Reversals* on page 27 and the section *Authorization Reversals* on page 89.**
Mastercard Bill Payments	**Added information about Cybersource support for Mastercard Bill Payment Services. See *Mastercard Bill Payments* on page 42 and the section *Mastercard Bill Payment Processing* on page 236.**
Mastercard Expert Monitoring Solutions	**Added *Mastercard Expert Monitoring Solutions* on page 42 and the section *Mastercard Expert Monitoring Solutions Processing* on page 240.**

23.03

Stored Credential Processing	**Updated the section *Stored Credentials* on page 46.**
Interchange Optimization	**Added *Interchange Optimization* on page 39.**
Japanese Payment Options	**Added *Japanese Payment Options* on page 40 and the section *Japanese Payment Options Processing* on page 193.**
Introduction to Payments	**Made changes to the section *Card Types* on page 21.**
Cash Advances with Credit Cards	**Added the section *Authorization for a Cash Advance with a Credit Card* on page 129.**

About This Guide

This section describes how to use this guide and where to find further information.

Audience and Purpose

This guide is written for application developers who want to use the REST API to integrate payment card processing into an order management system.
Implementing the Cybersource payment services requires software development skills. You must write code that uses the API request and response fields to integrate the credit card services into your existing order management system.
Visit the *Cybersource documentation hub* to find additional processor-specific versions of this guide and additional technical documentation.

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.

Customer Support

For support information about any service, visit the Support Center:
http://support.cybersource.com

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.

Introduction to Payments

This introduction provides the basic information that you will need to successfully process payment transactions. It also provides an overview of the payments industry and provides workflows for each process.
With Cybersource payment services, you can process payment cards (tokenized or non-tokenized), digital payments such as Apple Pay and Google Pay, and customer ID transactions. You can process payments across the globe and across multiple channels with scalability and security. Cybersource supports a large number of payment cards and offers a wide choice of gateways and financial institutions, all through one connection.
Visit the *Cybersource documentation hub* to find additional processor-specific versions of this guide and additional technical documentation.

Financial Institutions and Payment Networks

Financial institutions and payment networks enable payment services. These entities work together to complete the full payment cycle.

Merchant Financial Institutions (Acquirers)

A merchant financial institution, also known as an acquirer, offers accounts to businesses that accept payment cards. Before you can accept payments, you must have a merchant account from an acquirer. Your merchant account must be configured to process card-not-present, card-present, or mail-order/telephone-order (MOTO) transactions.
Each acquirer has connections to a limited number of payment processors. You must choose a payment processor that your acquirer supports.
You can expect your acquirer to charge these fees:

- Discount rates: your acquirer charges a fee and collects a percentage of every transaction. The combination of the fee and the percentage is called the discount rate. These charges can be bundled (combined into a single charge) or unbundled (charged separately).

- Interchange fees: payment networks, such as Visa or Mastercard, each have a base fee, called the interchange fee, for each type of transaction. Your acquirer and processor can show you ways to reduce this fee.
- Chargebacks: when cardholders dispute charges, you can incur chargebacks. A chargeback occurs when a charge on a customer's account is reversed. Your acquirer removes the money from your account and could charge you a fee for processing the chargeback.

Take these precautions to prevent chargebacks:

- Use accurate merchant descriptors so that customers can recognize the transactions on their statements.
- Provide good customer support.
- Ensure rapid problem resolution.
- Maintain a high level of customer satisfaction.
- Minimize fraudulent transactions.

If excessive chargebacks or fraudulant changes occur, these actions might be taken:

- You might be required to change your business processes to reduce the number chargebacks, fraud, or both.
- Your acquiring institution might increase your discount rate.
- Your acquiring institution might revoke your merchant account.

Contact your sales representative for information about products that can help prevent fraud.

Issuing (Customer) Financial Institutions

An issuing (customer) financial institution, also known as an issuer, provides payment cards to and underwrites lines of credit for their customers. The issuer provides monthly statements and collects payments. The issuer must follow the rules of the payment card companies to which they belong.

Payment Networks

Payment networks manage communications between acquiring financial institutions and issuing financial institutions. They also develop industry standards, support their brands, and establish fees for acquiring institutions.
Some payment networks, such as Visa and Mastercard, are trade associations that do not issue cards. Issuers are members of these associations, and they issue cards under license from the association.
Other networks, such as Discover and American Express, issue their own cards. Before you process cards from these companies, you must sign agreements with them.

Payment Processors

Payment processors connect with acquirers. Before you can accept payments, you must register with a payment processor. An acquirer might require you to use a payment processor with an existing relationship with the acquirer.

Your payment processor assigns one or more merchant IDs (MIDs) to your business. These unique codes identify your business during payment transactions.

Card Types

You can process payments with these card types:

- Co-badged cards
- Co-branded cards
- Credit cards
- Debit cards
- Prepaid cards
- Private label cards
- Quasi-cash

You can process payments with these card brands:

- Visa
- Mastercard
- JCB
- Discover
- American Express
- Diners Club
- China UnionPay

Co-Badged Cards

Co-badged cards are credit and debit cards that integrate two or more payment networks.

mada Co-Badged Cards

mada is Saudi Arabia's domestic payment network.
These mada co-badged debit cards are supported:

- Visa and mada
- Mastercard and mada

mada co-badged debit cards are processed as follows:

- Only domestic processing is supported in Saudi Arabia.
- Transactions are sent directly to the Saudi Arabia Monetary Authority (SAMA) for processing.
- Payer authentication is supported. Visa Secure is supported for co-badged Visa and mada cards. Mastercard Identity Check is supported for co-badged Mastercard and mada cards.
- For acquirers, the card type is identified as MD.

- In reports, the card type is identified as either Visa or Mastercard.
- Dual-message processing is not supported. Only single-message processing is supported.

Co-Branded Cards

Co-branded cards are credit cards that are branded with a merchant's logo, brand, or other identifier as well as the payment network logo. These cards are not limited for use at the branded merchant and can be used at any merchant that accepts credit cards.

Credit Cards

Cardholders use credit cards to borrow money from issuing banks to pay for goods and services offered by merchants that accept credit cards.

Debit Cards

A debit card is linked to a cardholder's checking account. A merchant who accepts the debit card can deduct funds directly from the account.

Prepaid Cards

Prepaid cards enable cardholders to pay for goods and services using money stored directly on the card.

Private Label Cards

Private label cards are issued by private companies. They enable cardholders to borrow money to pay for goods exclusively at the issuing company’s stores.

Quasi-Cash

Quasi-cash transactions involve instruments that are directly convertible to cash such as web wallets, travelers checks, cryptocurrency, and lottery tickets.

Transaction Types

This topic provides information about transaction types that are supported by your processor, such as card-present, card-not-present, and international transactions.

Card-Not-Present Transactions

When a customer provides a card number, but the card and the customer are not physically present at the merchant's location, the purchase is known as a card-not-present transaction. Typical card-not-present transactions are internet and phone transactions. Card-not-present transactions pose an additional level of risk to your business because the customer’s identification cannot be verified. You can reduce that risk by using features such as the Address Verification System (AVS) and Card Verification

Numbers (CVNs). The AVS and CVNs provide additional protection from fraud by verifying the validity of the customer's information and notifying you when discrepancies occur.

Card-Present Transactions

When a customer uses a card that is physically present in a retail environment, the purchase is known as a card-present transaction.

Authorizations with Card Verification Numbers

Card verification numbers (CVNs) are a required feature for the authorization service. The CVN is printed on a payment card, and only the cardholder can access it. The CVN is used in card-not-present transactions as a verification feature. Using the CVN helps reduce the risk of fraud.
CVNs are not included in payment card track data and cannot be obtained from a card swipe, tap, or dip.
CVNs must not be stored after authorization.

Important

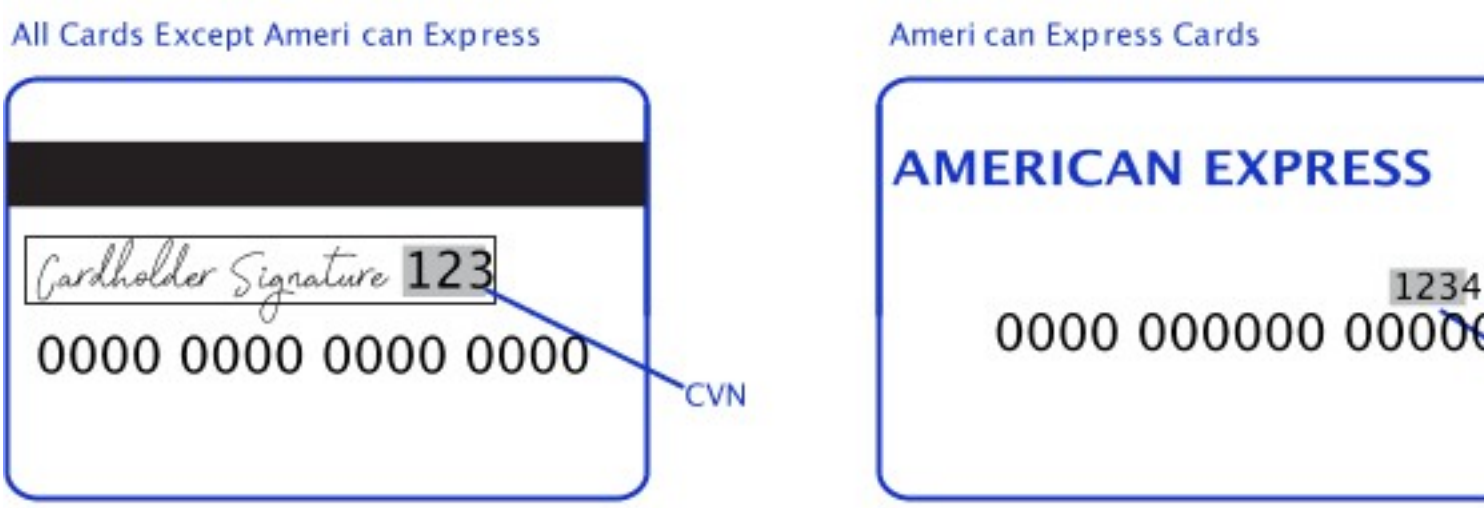
In Europe, Visa mandates that you not include a CVN for mail-order transactions and not record a CVN on any physical format such as a mail-order form.

Endpoint

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

CVN Locations and Terminology

For most cards, the CVN is a three-digit number printed on the back of the card, to the right of the signature field. For American Express, the CVN is a four-digit number printed on the front of the card above the card number.



CVN Locations

Each payment card company has its own name for the CVN value:

- American Express and Discover call it the Card Identification Number (CID).
- JCB calls it the Card Authentication Value (CAV2).
- Mastercard calls it the Card Validation Code (CVC2).
- Visa calls it the Card Verification Value (CVV2).

International Transactions

Consider these dynamic currency conversion and merchant remittance funding when processing international transactions.

Dynamic Currency Conversion

Dynamic Currency Conversion (DCC) is a service that enables users to conver the price of a transaction from a merchant's local currency to the customer's billing currency in real time. It is regulated by Mastercard and Visa.
The DCC Gateway service allows users to choose thier own service provider for DCC while complying with Mastercard and Visa payment processing rules for DCC transactions. The currency conversion is performed directly between the user and a third-party provider prior to authorizing a network compliant DCC transaction on their processor connection. Using the DCC Gateway option, you can comply with card mandates and other regulations while using a third-party currency conversion service.

Merchant Remittance Funding

You can request that the transaction proceeds be converted to another currency. Currency conversion uses a foreign exchange rate to calculate the conversion to the requested currency. The foreign exchange rate might be explicitly stated as a rate or implicitly stated as a transaction amount. The funded amount and can vary from day to day.

The foreign exchange rate might also include an increase for the foreign exchange risk, sales commissions, and handling costs.

Token Management Service

The Token Management Service (TMS) tokenizes, securely stores, and manages customer and payment data. TMS enables you to:

- Securely store a customer's payment details and their billing and shipping addresses.
- Create a network token of a customer's payment card.

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

- Customer — Stores the buyer's email address and the merchant's account ID for that buyer plus any other custom fields.
- Shipping Address — Stores a shipping address for a specific customer.
- Instrument Identifier — Stores either a payment card number or a bank account number and routing number
  This resource creates either:
  - An Instrument Identifier token using details of a payment card or an ACH bank account.
  - A payment network token using the details of a payment card; also uses the card expiration date and billing address, which are pass-through only fields.
- Payment Instrument — Stores a Payment Instrument using an Instrument Identifier token. It does not store the card number and cannot exist without an associated Instrument Identifier. It stores:
  - Card expiration date
  - Billing address

  You can also choose to store this information yourself instead and store only the card number or bank account and routing number in an Instrument Identifier object.
- Customer Payment Instrument — Creates and stores a payment instrument for a specific customer ID and an Instrument Identifier token.

TMS Features

- Create, retrieve, update, and delete tokens.
- Set a default payment instrument and shipping address for a customer.
- Process follow-on payment transactions with token IDs.
- Create and update tokens through bundled payment transactions.

Important

Due to mandates from the Reserve Bank of India, Indian merchants cannot store personal account numbers (PAN). Use network tokens instead. For more

information on network tokens, see the Network Tokenization section of the *Token Management Service Guide.*

Payment Services

Various services are use to process payments. These services enable customers to purchase goods and services, merchants to receive payments from the customer's accounts, merchants to provide refunds, and merchants to void transactions.

Authorizations

An authorization confirms that a payment card account holds enough funds to pay for a purchase. Authorizations can be made online or offline.

Online Authorizations

Online authorizations provide immediate confirmation of funds availability. The customer's financial institution also reduces the amount of credit available in the customer's account, setting aside the authorized funds for the merchant to capture at a later time. Authorizations for most payment cards are processed online. Typically, it is safe to start fulfilling the order when you receive an authorization confirmation.
An online authorization confirmation and the subsequent hold on funds expires after a specific length of time. Thus it is important to capture funds in a timely manner. The issuing bank sets the expiration time interval, but most authorizations expire within 5 to 7 days. The issuing bank does not inform Cybersource when an authorization confirmation expires. By default, the authorization information for each transaction remains in the Cybersource database for 180 days after the authorization date. To capture an authorization that expired with the issuing bank, you can resubmit the authorization request.

Offline Authorizations

Online transactions require an internet connection. In situations where the internet is not available, for example, due to an outage, merchants can continue to take credit card payments using offline transactions. An offline authorization is an authorization request for which you do not receive an immediate confirmation about the availability of funds. Offline authorizations have a higher level of risk than online transactions because they do not confirm funds availability or set aside the funds for later capture. Further, it can take up to 5 days to receive payment confirmations for offline transactions. To mitigate this risk, merchants may choose to fulfill orders only after receiving payment confirmation.

Payment Network Token Authorizations

You can integrate authorizations with payment network tokens into your existing order management system. For an incremental authorization, you do not need to include any payment network tokenization fields in the authorization request because Cybersource obtains the payment network tokenization information from the original authorization request.

Authorization Workflow

This image and description show the authorization workflow:

1. The customer purchases goods or service from the merchant using a payment card.
2. The merchant sends an authorization request to the acquiring (merchant) bank.
3. The acquiring (merchant) bank forwards the authorization request to the payment network.
4. The payment network forwards the authorization request to the issuer (customer) bank.
5. If funds are available, the issuer (customer) bank reserves the amount of the authorization request and returns an authorization approval to the payment network. If the issuer (customer) bank denies the request, it returns an authorization denial.
6. The payment network forwards the message to the acquiring (merchant) bank.
7. The acquiring (merchant) bank forwards the message to the merchant.

Authorization Reversals

An authorization reversal releases the hold that an authorization placed on a customer's payment card funds.

Each card-issuing financial institution has its own rules for deciding whether an authorization reversal succeeds or fails. When a reversal fails, contact the card-issuing financial institution to learn whether there is a different way to reverse the authorization.

If your processor supports authorization reversal after void (ARAV), you can reverse an authorization after you void the associated capture. If your processor does not support ARAV, you can use the authorization reversal service only for an authorization that has not been captured and settled.

An authorization reversal is a follow-on transaction that uses the request ID returned from an authorization. The main purpose of a follow-on transaction is to link two transactions. The request ID links the follow-on transaction to the original transaction. The authorization request ID is used to look up the customer's billing and account information in the Cybersource database. You are not required to include those fields in the full authorization reversal request. The original transaction and follow-on transaction are linked in the database and in the Business Center.

For processors that support debit cards and prepaid cards, the full authorization reversal service works for debit cards and prepaid cards in addition to credit cards.

Important

You cannot perform an authorization reversal if a transaction is in a review state, which can occur if you use a fraud management service. You must reject the transaction prior to authorization reversal. For more information, see the fraud management documentation in the Business Center.

Automatic Partial Authorization Reversals

Automatic partial authorization reversals are supported for:

- Credit cards
- Debit cards and prepaid cards.
- Quasi-cash.

If the capture amount is less than the authorization amount, Cybersource automatically performs a partial authorization reversal before it sends the capture request to the processor. The results of a successful partial authorization reversal are:

- The capture amount matches the new authorization amount at the payment card company.
- The hold on the unused credit card funds might be released. The issuing bank decides whether or not to release the hold on unused funds.
  Not all issuers act on a request for a partial authorization reversal. Therefore, Cybersource cannot guarantee that the funds will be released.

Credits

Credits are payment refunds from a merchant to the cardholder after a cardholder pays for a product or service and that payment is captured by the merchant. When a credit request is successful, the issuer transfers funds from the merchant bank (acquirer) account to the customer's account. It typically takes 2 to 4 days for the acquirer to transfer funds from your merchant account.

Warning

You should carefully control access to the credit service. Do not request this service directly from your customer interface. Instead, incorporate this service as part of your customer service process. This process reduces the potential for fraudulent transactions.

There are two basic types of credits: refunds and stand-alone credits.

Refunds

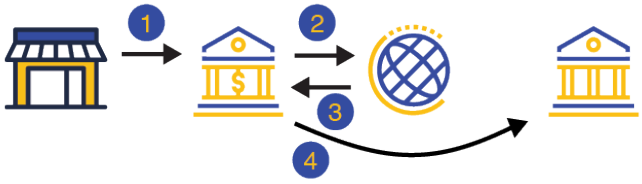
Refunds, also known as follow-on credits, use the capture request ID to link the refund to a specific transaction. This request ID is returned during the capture request (also known as a settlement) and is used in all subsequent refunds associated with the original capture. The request ID links the transaction to the customer’s billing and account information, so you are not required to include those fields in the credit request. However, when you combine a request for a refund with a request for another service, such as the tax calculation service, you must provide the customer’s billing and account information. Unless otherwise specified, refunds must be requested within 180 days of a settlement. You can request multiple refunds against a single capture. To perform multiple refunds, use the same request ID in each request.

Stand-Alone Credits

Stand-alone credits are not tied to an original transaction. Stand-alone credits do not have a time restriction, and they can be used to issue refunds more than 180 days after a transaction settlement.

Credit Workflow

A credit does not happen in real time. All of the credit requests for a day are typically placed in a file and sent to the processor as a single batch transaction. In most cases, the batch transaction is settled overnight.
This image and description show the credit workflow:



1. The merchant sends the credit request to Cybersource.
2. The order information is validated by Cybersource.
3. The credit request is sent to the acquirer.
4. The acquirer transfers the requested funds to the issuer.

Voids

A void cancels a capture or credit request that was submitted but not yet processed by the processor.
Capture and credit requests are usually submitted once a day. A void request is declined when the capture or credit request has already been sent to the processor.
After a void is processed, you cannot credit or capture the funds. You must perform a new transaction to capture or credit the funds. Further, when you void a capture, a hold remains on the authorized funds. If you are not going to re-capture the authorization, and if your processor supports authorization reversal after void (ARAV), you should request an authorization reversal to release the hold on the unused funds.
A void uses the capture or credit request ID to link the transactions. The authorization request ID is used to look up the customer's billing and account information, so there is no need to include those fields in the void request. You cannot perform a follow-on credit against a capture that has been voided.

Sales

A sale is a bundled authorization and capture. Some processors and acquirers require a sale transaction instead of using separate authorization and capture requests. For other processors and acquirers, you can request a sale instead of a separate authorization and capture when you provide the goods or services immediately after taking an order.
There are two types of sale processing: dual-message processing and single-message processing.

Dual-Message Processing

Dual-message processing is a two-step process. The authorization is processed first. If the authorization is successful, the capture is processed immediately afterward. The response includes the authorization and the capture information. If the authorization is declined, the capture is not processed, and the response message includes only the authorization information.

Partial Authorizations

All debit and prepaid card processors as well as a limited number of credit card processors support partial authorizations when dual-message processing is in place.

When partial authorization is enabled, the issuing financial institution can approve a partial amount when the balance on the card is less than the requested amount. When a partial amount is authorized, the capture is not processed. The merchant can then use a second card to cover the balance, adjust the total cost, or void the transaction.

Single-Message Processing

Single-message processing treats the authorization and capture as a single transaction. There are important differences between dual-message processing and single-message processing:

- Single-message processing treats the request as a full-financial transaction, and with a successful transaction, funds are immediately transferred from the customer account to the merchant account.
- Authorization and capture amounts must be the same.
- Some features cannot be used with single-message processing.

Payment Features

You can apply features to different payment services to enhance the customer payment processing experience. This section includes an overview of these features:

Card-Present Authorizations

For card-present transactions, the presence of the payment card is established during the authorization service. These are the basic types of card-present authorizations:

- EMV authorization: Authorization that is based on the EMV chip embedded in the cardholder's card.
- Magnetic stripe authorization: Authorization that is based on the magnetic stripe on the back of the cardholder's card.
- Hand-keyed authorization: Authorization that is based on you manually entering the card information into the payment terminal.

After you complete a card-present authorization, using these follow-on services enables you to complete the full payment workflow:

- Capture
- Contact EMV capture
- Stand-alone credit
- Authorization reversal
- Void

Related Information

- See the *Card Present Connect | Retail Integration Guide* for more information about the retail transactions.
- See *Card Present Connect | Retail Processing* on page 113 for information that shows you how to process retail payments.
- See the *Card Present Copnnect | Mass Transit Developer Guide* for more information about the mass transit transactions.
- See *Card Present Connect | Mass Transit Processing* on page 136 for information that shows you how to process mass transit payments.

Debit and Prepaid Card Payments

Debit cards are linked to a cardholder's checking account. A merchant who accepts the debit card can deduct funds directly from the linked cardholder's account.
You can process debit cards using these services:

- Credit card services
- PIN debit services
- Partial authorizations, which are a special feature available for debit cards
- Balance inquiries, which are a special feature available for debit cards

Requirements

In Canada, to process domestic debit transactions on Visa Platform Connect with Mastercard, you must contact customer support to have your account configured for this feature.

Related Information

- See *Standard Payment Processing* on page 50 for information that shows you how to use credit card services.
- See *Debit and Prepaid Card Processing* on page 162 for information that shows you how to process authorizations that use a debit or prepaid card.

Airline Data

Airline data processing goes beyond basic payment transactions by allowing you to process specific travel data. This requires you to submit additional information, such as:

- Carrier
- Departure Date
- Destination Airport
- Purchase Date
- Originating Airport
- Ticket Class
- Trip Legs

Supported Card Types

- American Express
- Discover
- Mastercard
- Visa

Supported Acquirers

These Visa Platform Connect acquirers are supported for airline data processing:

- Agricultural Bank of China (ABC)
- Ahli United Bank in Bahrain
- Arab African International Bank (AAIB)
- Asia Commercial Bank (ACB)
- Auckland Savings Bank (ASB)
- Axis Bank Ltd. of India
- Bangkok Bank Ltd.
- Bank Muscat of Oman
- Bank of Ayudhya (BAY)
- Bank of China (BOC)
- Bank of Communications
- Bank Sinarmas (Omise Ltd.)
- Banque Pour Le Commerce Exterieur Lao (BCEL)
- Barclays Bank Mauritius Ltd.
- Barclays Bank Botswana

- Barclays Bank of Ghana Ltd., Barclays Bank of Tanzania Ltd., and Barclays Bank of Uganda Ltd.
- Barclays Bank of Kenya
- Barclays Bank of Zambia
- Barclays Bank Seychelles
- BC Card Co., Ltd.
- BLOM Bank
- Cathay United Bank (CUB)
- Citibank Hongkong and Macau
- Citibank Singapore Ltd.
- Commercial Bank of Qatar
- CrediMax (Bahrain)
- CTBC Bank Ltd.
- FirstRand Bank
- Global Payments Asia Pacific
- Habib Bank Ltd. (HBL)
- HDFC Bank Ltd. of India
- I&M Bank
- ICICI of India
- Korea Exchange Bank (KEB)
- Mashreq
- National Bank of Abu Dhabi (NBAD)
- National Bank of Kuwait (NBK)
- National Commercial Bank
- Network International
- Overseas Chinese Banking Corp (OCBC)
- Promerica in Honduras and Nicaragua
- Qatar National Bank (QNB Group)
- Raiffeisenbank
- Rosbank
- Taishin Bank Ltd.
- United Overseas Bank (UOB) in Singapore and Vietnam
- United Overseas Bank (UOB) in Thailand
- Vietcombank
- VTB24
- Wing Lung Bank

Requirement

When you are ready to go live with airline data processing, contact Cybersource Customer Support to have your account configured to process airline data. If your account is not enabled, and you try to send airline transactions, you will receive an error for invalid data.

Related Information

- See *Airline Data Processing* on page 174 for information that shows you how to process payments that include airline data.

Cybersource Airline Data Processing

Cybersource does not store airline data. Instead, it functions as a pass-through service for the data. Cybersource enforces only the minimal level of field validation.
When you request an airline service, Cybersource responds with certain fields and values to indicate whether the airline data was processed. The response fields for each service are:

- Authorization: **processingInformation.enhancedDataEnabled**
- Capture: **processingInformation.enhancedDataEnabled**
- Credit: **processingInformation.enhancedDataEnabled**

The possible values for the response fields are:

- `Y`: the airline data was included in the request to the processor.
- `N`: the airline data was not included in the request to the processor.

Cybersource temporarily disables your account's airline data processing capability and contacts you if your airline data transactions produce batching errors when the information is sent to the processor. If this happens, your request is not rejected, but you receive one of the above listed fields with the `N` value in the response indicating that airline data in the request has been ignored and not sent to the processor.

Airline Travel Legs

This section shows you how to process an airline transaction with travel legs for this processor:

- Visa Platform Connect

Using Travel Legs

To include travel legs in an airline transaction, include one or more travel legs in the **legs[]** array.
For example, these three travel legs are valid:

```
"travelInformation": {
  "transit": {
   "airline": {
    "legs": [
     {
      "carrierCode": "XX"
     },
     {
      "carrierCode": "XZ"
     },
     {
      "carrierCode": "XX"
```

```
    }
   ]
  }
```

Important

If you skip a number, Cybersource ignores the legs that follow the skipped number.

Travel Leg Limitations

Some processors limit the amount of travel legs for each trip based on card type. For more information, see the Leg Limitations table in the capture section of the processor you are using.

Airline Data Reference Information

This section contains reference information that is useful when using Airline Data.

Airline Document Type Codes

To indicate the purpose of a purchase, set the **travelInformation.transit.airline.documentType** field to a value listed in the Code column.

Airline Document Type Codes

Code	Description
01	Passenger ticket
02	Additional collection
03	Excess baggage
04	Miscellaneous charge order (MCO) or prepaid ticket authorization
05	Special service ticket
06	Supported refund
07	Unsupported refund
08	Lost ticket application
09	Tour order voucher
10	Ticket by mail
11	Undercharge adjustment
12	Group ticket
13	Exchange adjustment
14	SPD or air freight
15	In-flight adjustment
16	Agency passenger ticket

Code	Description
17	Agency tour order or voucher
18	Agency miscellaneous charge order (MCO)
19	Agency exchange order
20	Agency group ticket
21	Debit adjustment for duplicate refund or use
22	In-flight merchandise order
23	Catalogue merchandise order
24	In-flight phone charges
25	Frequent flyer fee or purchase
26	Kennel charge
27	Animal transportation charge
28	Firearms case
29	Upgrade charge
30	Credit for unused transportation
31	Credit for class of service adjustment
32	Credit for denied boarding
33	Credit for miscellaneous refund
34	Credit for lost ticket refund
35	Credit for exchange refund
36	Credit for overcharge adjustment
37	Credit for multiple Unused tickets
38	Exchange order
39	Self-service ticket
41	In-flight duty-free purchase
42	Senior citizen discount booklets
43	Club membership fee
44	Coupon book
45	In-flight charges
46	Tour deposit
47	Frequent flyer overnight delivery charge

Code	Description
48	Frequent flyer fulfillment
49	Small package delivery
50	Vendor sale
51	Miscellaneous taxes or fees
52	Travel agency fee
60	Vendor refund or credit
64	Duty free sale
65	Preferred seat upgrade
66	Cabin upgrade
67	Lounge or club access or day pass
68	Agent assisted reservation or ticketing fee
69	Ticket change or cancel fee
70	Trip insurance
71	Unaccompanied minor
72	Standby fee
73	Curbside baggage
74	In-flight medical equipment
75	Ticket or pass print fee
76	Checked sporting or special equipment
77	Dry ice fee
78	Mail or postage fee
79	Club membership fee or temporary trial
80	Frequent flyer activation or reinstatement
81	Gift certificate
82	Onboard or in-flight prepaid voucher
83	Optional services fee
84	Advance purchase for excess baggage
85	Advance purchase for preferred seat upgrade
86	Advance purchase for cabin upgrade
87	Advance purchase for optional services

Code	Description
88	Wi-Fi
89	Packages
90	In-flight entertainment or internet access
91	Overweight bag fee
92	Sleep sets
93	Special purchase fee

Ancillary Service Category Codes

To indicate the service provided in an ancillary purchase, set the **travelInformation.transit.airline.ancillaryInformation.service[].categoryCode** and **travelInformation.transit.airline.ancillaryInformation.service[].subCategoryCode** fields to a value listed in the Ancillary Service Category Code column.

Ancillary Service Category Codes

Ancillary Service Category Codes	Description
BF	Bundled service
BG	Baggage fee
CF	Change fee
CG	Cargo
CO	Carbon offset
FF	Frequent flyer
GF	Gift card
GT	Ground transport
IE	In-flight entertainment
LG	Lounge
MD	Medical
ML	Meal or beverage
OT	Other
PA	Passenger assist fee
PT	Pets
SA	Seat fees
SB	Standby
SF	Service fee

Ancillary Service Category Codes	Description
ST	Store
TS	Travel service
UN	Unaccompanied travel
UP	Upgrades
WI	Wi-Fi

Interchange Optimization

Interchange fees are per-transaction transfer fees charged by your acquirer. The fee amount is based in part on the transaction amount that the acquirer submits to the payment network for clearing and settlement. Interchange optimization can help to reduce these fees for card-present transactions.

Payment Cards Supported with Interchange Optimization

- Mastercard
- Visa

Automatic Authorizations

Interchange optimization works by automatically performing additional authorization transactions for two types of card-not-present scenarios.

Automatic Authorization Refresh	**If a capture request occurs more than 6 days after the date of the original authorization, the processor automatically obtains a fresh authorization for the capture amount.**
Automatic Partial Authorization Reversal	**If the capture does not need a fresh authorization but the capture amount is less than the authorization amount, the processor automatically performs a partial authorization reversal. The reversal releases the hold on unused credit card funds and ensures that the settlement amount matches the authorization amount.**

How Interchange Optimization Transactions are Tracked

To find out when the processor performed automatic authorizations, see the daily processor report.

Limitations

- Interchange optimization does not apply to transactions in which the payment card is present at the merchant's physical place of business.
- Interchange optimization is not supported with incremental authorizations.

Requirement

Contact customer support to enable interchange optimization for your account.

Related Information

Merchant Financial Institutions (Acquirers) **on page 19**

Japanese Payment Options

Japanese payment options (JPO) extend the Cybersource payment card processing features to support payment methods used only in Japan. Japanese issuers, cardholders, merchants, and acquirers recognize payment methods that clarify the nature of a payment. JPO provides for more fine-grained identification of one-time payments and installment payments. You can offer your customers JPO payment methods that they select at the time of purchase.
JPO supports these payment methods:

- Single payment
- Bonus payment
- Installment payment
- Revolving payment
- Combination of bonus payment and installment payment

Important

Requests with Japanese payment options are accepted independently of your agreements with acquirers. When you submit a request with one of these payment options but do not have the necessary contracts and agreements in place, an error might not occur until the acquirer processes the settlement file.

For more information about the Japanese payment options, contact Customer Support of Cybersource KK (Japan).

Payment Cards Supported with JPO

JPO is supported for the Sumitomo Mitsui Card Co. acquirer with transactions that use Visa payment cards issued in Japan.

Services Supported with JPO

Authorization service.

Requirements

- You have signed a contract with your acquirer.
- You have contacted your account provider for details about contracts and funding cycles. The funding cycle could differ when using JPO.
- Card holders who want to use JPO have signed a contract with an issuing bank.
- You have confirmed payment option availability with your account provider and card holder before implementing one of these payment options.

Related Information

- See *Japanese Payment Options Processing* on page 193 for information that shows you how to process payments using JPO.

Level II and Level III Data

For business to business customers, Level II and Level III processing can provide lower interchange rates in exchange for providing more information during a transaction. Support for Level II and Level III data processing is processor and card specific.

Level II Data

Level II cards, which are also called Type II cards, provide customers with additional information on their credit card statements about their purchases. Level II cards enable customers to easily track the amount of sales tax they pay and to reconcile transactions with a unique customer code. There are two categories of Level II cards:

- Business/corporate cards are given by businesses to employees for business-related expenses such as travel and entertainment or for corporate supplies and services.
- Purchase/procurement cards are used by businesses for expenses such as supplies and services. These cards are often used as replacements for purchase orders.

Level III Data

You can provide Level III data for purchase/procurement cards, which are used by businesses for expenses such as supplies and services. These cards are often used as replacements for purchase orders. The Level III data is forwarded to the company that made the purchase. It enables the company to manage its purchasing activities.

Related Information

- See *Level II Processing* on page 213 for information that shows you how to process transactions that include Level II data.
- See *Level III Processing* on page 223 for information that shows how to process transactions that include Level III data.

Mastercard Bill Payments

In Brazil, you can participate in a Mastercard Bill Payment program. If your account is enrolled in the program, your customers can use their Mastercard payment cards to make payments on their outstanding bills.

> **Important**
>
> A Mastercard card payment at the point of sale (POS) when goods or services are purchased is not part of the Mastercard Bill Payment program.

When you send an authorization request for a Mastercard Bill Payment, include the API field that specifies the bill payment type.

Limitation

The Mastercard Bill Payment program supports only bills paid in Brazil using Mastercard payments cards with Visa Platform Connect.

Requirements

Sign up with Mastercard to participate in their bill payment program.

Related Information

- See *Mastercard Bill Payment Processing* on page 236 for information that shows you how to process Mastercard Bill Payments.

Mastercard Expert Monitoring Solutions

Mastercard Expert Monitoring Solutions provides a predictive, behavior-based fraud score in real time during authorizations for card-not-present transactions. The score indicates the likelihood that the requested transaction is fraudulent and the type of fraud that is suspected.

To assign the fraud score for a transaction, Mastercard compares the customer's transaction data to their transaction behavior history and to a regional card-not-present fraud detection model. The resulting score is returned in the body of the response message.

Limitations

This feature is supported on Mastercard Payment cards issued in the US only.

This feature is supported with Visa Platform Connect only.

Requirement

Contact customer support to enable Mastercard Expert Monitoring Solutions for your account.

Important

After this feature is enabled for your account, Mastercard returns a fraud score for all your card-not-present authorization requests for Mastercard payment cards issued in the US.

Related Information

- See *Mastercard Expert Monitoring Solutions Processing* on page 240 for information that shows you how to obtain the transaction fraud score determined by Mastercard Expert Monitoring Solutions.

Payer Authentication

Payer authentication is run before a transaction is submitted for authorization. Most of the time payer authentication is bundled with authorization so that after payer authentication happens, the transaction is automatically submitted for authorization. Payer authentication and authorization can be configured to occur as separate operations. This section shows you how to run payer authentication as a separate process and pass the payer authentication data when seeking authorization for a transaction. Payer authentication consists of a two-step verification process that adds an extra layer of fraud protection during the payment process. During transactions, the transaction device, location, past purchasing habits, and other factors are analyzed for indications of fraud. This process collects customer data during the transaction from at least two of these three categories:

- Something you have: A payment card or a payment card number
- Something you know: A password or pin
- Something you are: Facial recognition or fingerprint

Each of these payment card companies has its own payer authentication product:

- American Express: SafeKey
- Discover: ProtectBuy
- JCB: J/Secure
- Mastercard: Identity Check
- Visa: Visa Secure

Payer authentication can be used to satisfy the Strong Customer Authentication (SCA) requirement of the Payment Services Directive (PSD2). SCA applies to the European Economic Area (EEA) and the United Kingdom. SCA requires banks to perform additional checks when customers make payments to confirm their identity.

Related Information

- See the *Payer Authentication Developer Guide* for more information about payer authentication.

- See *Payer Authentication Processing* on page 245 for information about how to process payments with payer authentication.

Relaxed Requirements for Address Data and Expiration Date in Payment Transactions

With relaxed requirements for address data and the expiration date, not all standard payment request fields are required. It is your responsibility to determine whether your account is enabled to use this feature and which fields are required.

Related Information

- See *Relaxed Requirements for Address Data and Expiration Date in Payment Transactions* on page 264 for information about how to process payments with relaxed requirements for address data and expiration date.

Split Shipments

Split shipments enable you to split an order into multiple shipments with multiple captures. You can use this feature when a customer orders a product that is not yet available, or when one or some products are available but not all. You are able to request multiple partial captures for one authorization, multiple authorizations and one capture, or an authorization and a sale.
Cybersource provides the split shipment services for authorizations and captures. There are three scenarios and actions you can take:

- Multiple authorizations—Request more than one authorizations; when the order is placed for the unavailable product and after the product becomes available to ship.
- Multiple partial captures—Request an authorization, and then request multiple partial captures for the amount of the products you ship. When the remaining product becomes available, ship it and request another capture.
- Multiple authorizations with multiple partial captures—Request more than one authorizations and captures when all the products in the order are not available for immediate shipment. After the other products become available, request another authorization, and then a capture when you ship the remaining product.

How Split Shipments Transactions are Linked

All transactions for a split shipment are linked together in the Business Center and in reports. When you split an order into multiple shipments with multiple partial captures, Cybersource requests the additional authorizations for you.

Obtaining the Status of a System-Generated Authorization

Important

> A system-generated authorization is not performed in real time. The response message that you receive indicates that the request was received, not whether it was approved or declined.

A system-generated authorization can be declined for the same reasons that a regular authorization can be declined. Cybersource recommends you use one of following methods to obtain the status of the system-generated authorization request before shipping the product:

- Business Center—Use the capture request ID to search for the follow-on capture. The details for all related transactions are displayed on the Transaction Details page. It can take a maximum of 6 hours for the status of the system-generated authorization request to be available.
- Transaction Detail API—You must use version 1.3 or later of the report and include the parameter **includeExtendedDetail** in your query. It can take a maximum of 6 hours for the status of the system-generated authorization request to be available.
- Transaction Exception Detail Report—Cybersource recommends you use this report on a daily basis to identify transactions that were declined.

Additional Authorizations

When you need an additional authorization for an order, you can use the **link-to-request** field to link follow-on authorizations to the original authorization in addition to the basic fields required for every authorization request. The follow-on authorization is linked to the original authorization in the Business Center and in reports. The captures for these authorizations are also linked to the original authorization in the Business Center and in reports.

For an additional authorization on a processor that supports merchant-initiated transactions, the authorization request must include the subsequent authorization fields that are required for merchant-initiated transactions.

Additional Captures

When you need an additional capture for an order, Cybersource performs a system-generated authorization for additional capture requests using the payment data from the original authorization. The system-generated authorization is linked to the original authorization in the Business Center and in reports. The captures are linked to the authorizations in the Business Center and in reports through the request IDs as with any capture.

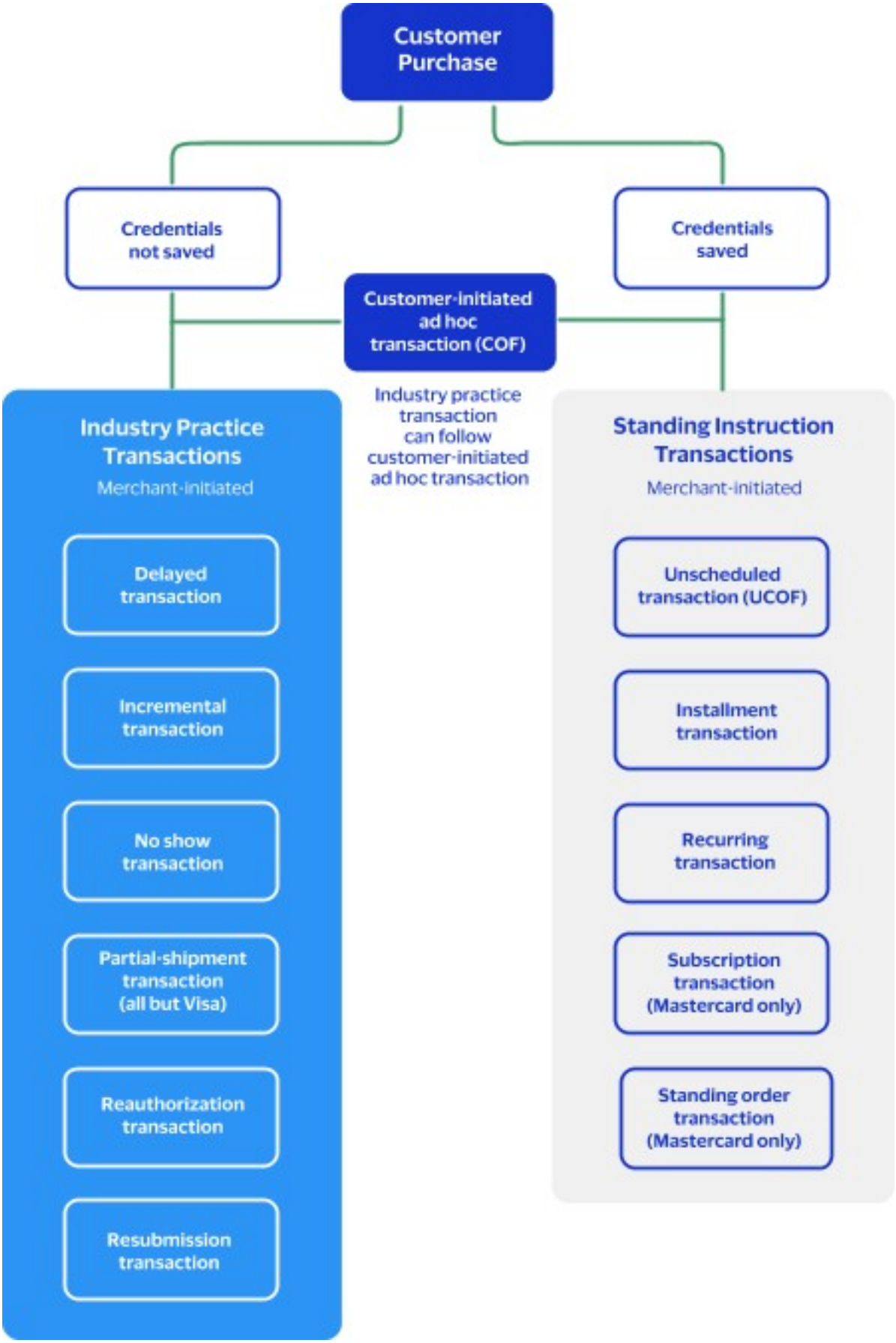
Related Information

- See *Authorizing a Sale for a Product Not Yet Available* on page 267 for guidelines on how to process a payment when a product is not available.
- See *Processing an Authorization and Two Captures for Multiple Products* on page 271 or *Processing Two Authorizations and a Capture for Multiple Products* on page 269 for guidelines on how to process payments for multiple products.

Stored Credentials

Credentialed transactions are transactions that involve either storing a customer's payment credentials for future transactions or using a customer's already stored payment credentials. When processing a credentialed transaction, you must indicate the type of credentialed transaction and the reason for the transaction.
There are several types of credentialed transactions:

- Customer-Initiated Transactions (CITs): Any transaction a customer is actively participating in such as making a card-present payment, completing an online checkout, or by using a stored credential.
- Merchant-Initiated Transactions (MITs): Any transaction a merchant initiates without the customer's participation such as an industry practice transaction or a standing instruction transaction.
    - Industry Practice Transactions: MITs that are performed as subsequent transactions to a CIT because the initial transaction could not be completed in one transaction. Not every industry practice transaction involves a stored credential. If a stored credential is used only for one transaction, that transaction is not considered a credentialed transaction.
    - Standing Instruction Transactions: MITs that are performed to follow agreed-upon instructions from the customer for the provision of goods and services.



MIT Types

Supported Services

These are the supported merchant-initiated services:

- Delayed Authorization
- Incremental Transactions
- Installment Transactions
- Mastercard Standing Order Transactions
- Mastercard Subscription Transactions
- No-Show Transactions
- Reauthorization
- Recurring Transactions
- Resubmission
- Unscheduled Credentials-on-File Transactions

Related Information

- See *Processing Payments Using Credentials* on page 274 for information that shows you how to process transactions using credentials.

Token Management Service

When using tokens, personally identifiable information (PII) including the primary account numbers (PANs) can be replaced with unique tokens. These tokens do not include the PII data, but act as a stand-in 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. The Token Management Service (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.

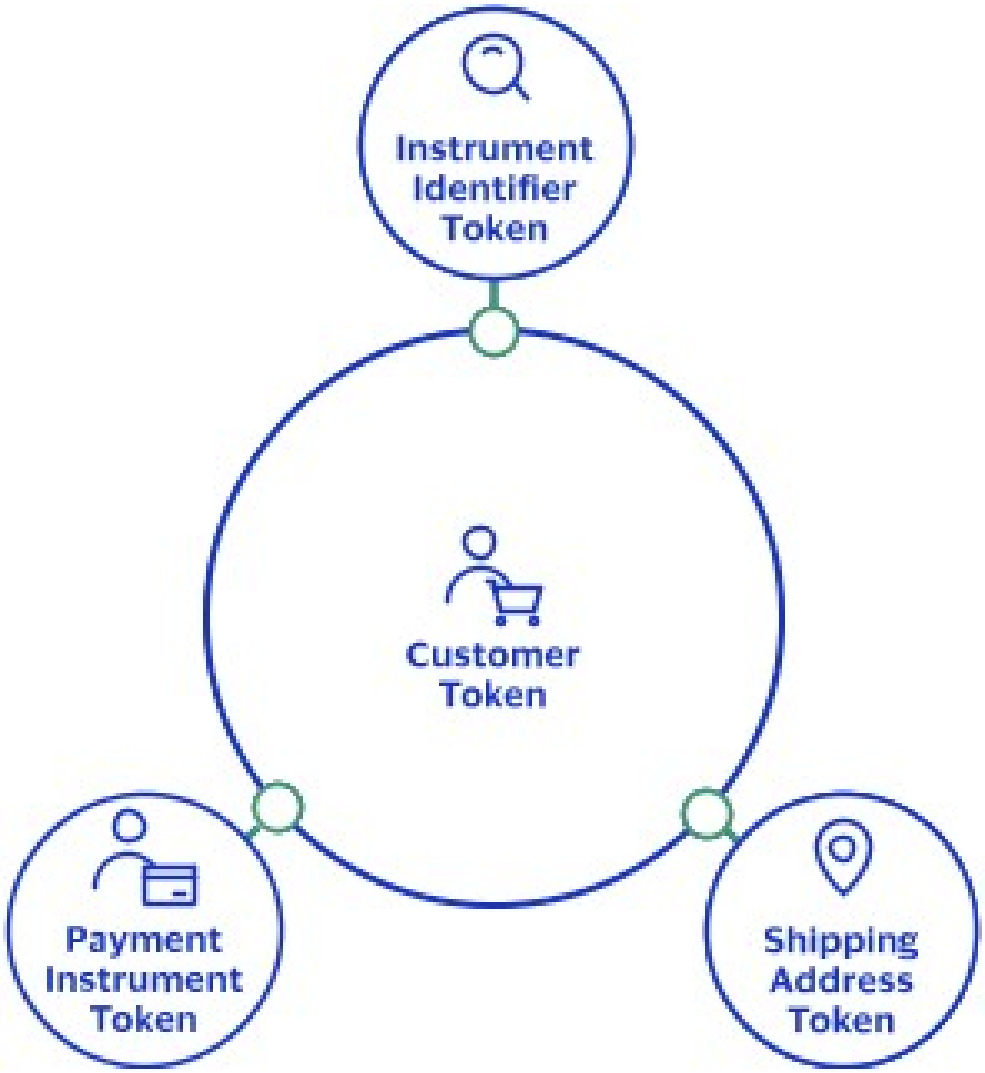
Important

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

Related Information

- See the *Token Management Service Developer Guide* for more information about the TMS.
- See *Token Management Service Processing* on page 421 for information that shows you how to process payments using the TMS.

Standard Payment Processing

This section shows you how to process various authorization, capture, credit, and sales transactions.

Additional Resources for Processing Payments

For more information, see these guides:

- *Payouts Developer Guide*
- *API field reference guide for the REST API*
- Github repositories: *https://github.com/Cybersource?q=rest-sample&type=all&language=&sort=*

Basic Authorizations

This section provides the information you need in order to process a basic authorization.

Endpoint

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

Declined Authorizations

If an authorization is declined, you can use response categories to help you decide whether to retry or block a declined transaction. These response fields provide additional information:

- **paymentInsightsInformation.responseInsights.category**

- **paymentInsightsInformation.responseInsights.categoryCode**

Category codes have possible values (such as `01`) each of which corresponds to a category that contains a description.
You cannot retry this category code and category:

- `01 ISSUER_WILL_NEVER_APPROVE`

For these values, you can retry the transaction a maximum of 15 times over a period of 30 days:

- `02 ISSUER_CANNOT_APPROVE_AT_THIS_TIME`
- `03 ISSUER_CANNOT_APPROVE_WITH_THESE_DETAILS`: Data quality issue. Revalidate data prior to retrying the transaction.
- `04 GENERIC_ERROR`
- `97 PAYMENT_INSIGHTS_INTERNAL_ERROR`
- `98 OTHERS`
- `99 PAYMENT_INSIGHTS_RESPONSE_CATEGORY_MATCH_NOT_FOUND`

Required Fields for Processing a Basic Authorization

Use these required fields for processing a basic authorization.

Important

When using relaxed requirements for address data and the expiration date, not all fields in this list are required. It is your responsibility to determine whether your account is enabled to use this feature and which fields are required. For details about relaxed requirements, see *Relaxed Requirements for Address Data and Expiration Date in Payment Transactions* on page 264.

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

Related Information

- *API field reference guide for the REST API*

Country-Specific Required Fields for Processing a Basic Authorization

Use these country-specific required fields to process a basic authorization.

Argentina

merchantInformation.taxId	**Required for Mastercard transactions.**
merchantInformation.transactionLocalDateTime	**Required in Argentina when the time zone is not included in your account. Otherwise, this field is optional.**

Brazil

paymentInformation.card.sourceAccountType	**Required for combo card transactions.**
paymentInformation.card.sourceAccountTypeDetailed	**Required for combo card line-of-credit and prepaid-card transactions.**

Chile

merchantInformation.taxId	**Required for Mastercard transactions.**

Paraguay

merchantInformation.taxId	**Required for Mastercard transactions.**

Saudi Arabia

processingInformation.authorizationOptions.authorizationMode	**Required only for merchants in Saudi Arabia.**

Taiwan

paymentInformation.card.hashedNumber	**Required only for merchants in Taiwan.**

Related Information

- *API field reference guide for the REST API*

REST Interactive Example: Processing a Basic Authorization

Simple Authorization(Internet)

Live Console URL: *https://developer.cybersource.com/api-reference-assets/index.html#payments_payments_process-a-payment*

REST Example: Processing a Basic Authorization

Endpoint:

- Production: POST https://api.cybersource.com/pts/v2/payments
- Test: POST https://apitest.cybersource.com/pts/v2/payments

Request

```
{
  "orderInformation": {
    "billTo": {
      "country": "US",
      "lastName": "Kim",
      "address1": "201 S. Division St.",
      "postalCode": "48104-2201",
      "locality": "Ann Arbor",
      "administrativeArea": "MI",
      "firstName": "Kyong-Jin",
      "email": "test@cybs.com"
    },
    "amountDetails": {
      "totalAmount": "100.00",
      "currency": "usd"
    }
  },
  "paymentInformation": {
    "card": {
      "expirationYear": "2031",
      "number": "4111111111111111",
      "expirationMonth": "12",
      "type": "001"
    }
  }
}
```

Response to a Successful Request

```
{
 "_links" : {
  "authReversal" : {
   "method" : "POST",
   "href" : "/pts/v2/payments/6461731521426399003473/reversals"
  },
  "self" : {
   "method" : "GET",
   "href" : "/pts/v2/payments/646173152142639900 3473"
  },
```

```
    "capture" : {
      "method" : "POST",
      "href" : "/pts/v2/payments/6461731521426399003473/captures"
    }
  },
  "clientReferenceInformation" : {
    "code" : "1646173152047"
  },
  "id" : "6461731521426399003473",
  "orderInformation" : {
    "amountDetails" : {
      "authorizedAmount" : "100.00",
      "currency" : "usd"
    }
  },
  "paymentAccountInformation" : {
    "card" : {
      "type" : "001"
    }
  },
  "paymentInformation" : {
    "tokenizedCard" : {
      "type" : "001"
    },
    "card" : {
      "type" : "001"
    }
  },
 "paymentInsightsInformation" : {
    "responseInsights" : {
      "categoryCode" : "01"
    }
  },
  "processorInformation" : {
    "systemTraceAuditNumber" : "862481",
    "approvalCode" : "831000",
    "merchantAdvice" : {
      "code" : "01",
      "codeRaw" : "M001"
    },
    "responseDetails" : "ABC",
    "networkTransactionId" : "016153570198200",
    "consumerAuthenticationResponse" : {
      "code" : "2",
      "codeRaw" : "2"
    },
    "transactionId" : "016153570198200",
    "responseCode" : "00",
    "avs" : {
      "code" : "Y",
      "codeRaw" : "Y"
    }
  },
  "reconciliationId" : "6461731521426399003473",
  "status" : "AUTHORIZED",
  "submitTimeUtc" : "2022-03-01T22:19:12Z"
```

```
}
```

Response to a Declined Request

```
{
 "clientReferenceInformation": {
  "code": "TC50171_3"
 },
 "errorInformation": {
  "reason": "PROCESSOR_ERROR",
  "message": "Invalid account"
 },
 "id": "6583553837826789303954",
 "paymentInsightsInformation": {
  "responseInsights": {
   "categoryCode": "01",
   "category": "ISSUER_WILL_NEVER_APPROVE"
  }
 },
 "pointOfSaleInformation": {
  "amexCapnData": "1009S0600100"
 },
 "processorInformation": {
  "systemTraceAuditNumber": "004544",
  "merchantNumber": "1231231222",
  "networkTransactionId": "431736869536459",
  "transactionId": "431736869536459",
 "responseCode": "111",
  "avs": {
   "code": "Y",
   "codeRaw": "Y"
  }
 },
 "status": "DECLINED"
}
```

Authorizations with Line Items

This section shows you how to process an authorization with line items.

The main difference between a basic authorization and an authorization that includes line items is that the **orderInformation.amountDetails.totalAmount** field, which is included in a basic authorization, is substituted with one or more line items that are included in a **lineItem[]** array.

Fields Specific to this Use Case

These fields are required for each line item that you use:

orderInformation.lineItems[].unitPrice

orderInformation.lineItems[].quantity

orderInformation.lineItems[].productCode	
orderInformation.lineItems[].productSku	Optional when **item_#_productCode** is set to `default`, `shipping_only`, `handling_only`, or `shipping_and_handling`
orderInformation.lineItems[].productName	Optional when **item_#_productCode** is set to `default`, `shipping_only`, `handling_only`, or `shipping_and_handling`

At a minimum, you must include the **orderInformation.lineItems[].unitPrice** field in order to include a line item in an authorization. When this field is the only field included in the authorization, the system sets:

- **orderInformation.lineItems[].productCode**: `default`
- **orderInformation.lineItems[].quantity**: `1`

For example, these three line items are valid.

```
"orderInformation": {
 "lineItems": [
  {
   "unitPrice": "10.00"
  },
  {
   "unitPrice": "5.99",
   "quantity": "3",
   "productCode": "shipping_only"
  },
  {
   "unitPrice": "29.99",
   "quantity": "3",
   "productCode": "electronic_good",
   "productSku": "12384569",
   "productName": "receiver"
  }
 ]
}
```

Endpoint

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

Required Fields for Processing an Authorization with Line Items

Use these required fields for processing an authorization that includes line items.

Important

When using relaxed requirements for address data and the expiration date, not all fields in this list are required. It is your responsibility to determine whether your account is enabled to use this feature and which fields are required. For details

about relaxed requirements, see *Relaxed Requirements for Address Data and Expiration Date in Payment Transactions* on page 264.

orderInformation.amountDetails.currency

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

Related Information

- *API field reference guide for the REST API*

Country-Specific Required Fields for Processing an Authorization with Line Items

Use these country-specific required fields to process a process an authorization with line items.

Argentina

merchantInformation.taxId	**Required for Mastercard transactions.**
merchantInformation.transactionLocalDateTime	**Required in Argentina when the time zone is not included in your account. Otherwise, this field is optional.**

Brazil

paymentInformation.card.sourceAccountType	**Required for combo card transactions.**
paymentInformation.card.sourceAccountTypes	**Required for combo card line-of-credit and prepaid-card transactions.**

Chile

merchantInformation.taxId	**Required for Mastercard transactions.**

Paraguay

merchantInformation.taxId	**Required for Mastercard transactions.**

Saudi Arabia

processingInformation.authorizationOptions.transactionMode	**Required only for merchants in Saudi Arabia.**

Related Information

- *API field reference guide for the REST API*

REST Example: Processing an Authorization with Line Items

Endpoint:

- Production: POST https://api.cybersource.com/pts/v2/payments
- Test: POST https://apitest.cybersource.com/pts/v2/payments

Request

```
{
 "currencyConversion": {
  "indicator": "Y"
 },
 "paymentInformation": {
  "card": {
   "number": "4111111111111111",
   "expirationMonth": "12",
   "expirationYear": "2031"
  }
 },
 "orderInformation": {
  "amountDetails": {
   "currency": "USD",
   "exchangeRate": ".91",
   "originalAmount": "107.33",
   "originalCurrency": "eur"
  },
  "billTo": {
   "firstName": "John",
   "lastName": "Doe",
   "address1": "1 Market St",
   "locality": "san francisco",
   "administrativeArea": "CA",
   "postalCode": "94105",
   "country": "US",
   "email": "test@cybs.com"
  },
```

```
   "lineItems": [
    {
     "unitPrice": "10.00"
    },
    {
     "unitPrice": "5.99",
     "quantity": "3",
     "productCode": "shipping_only"
    },
    {
     "unitPrice": "29.99",
     "quantity": "3",
     "productCode": "electronic_good",
     "productSku": "12384569",
     "productName": "receiver"
    }
   ]
  }
}
```

Response to a Successful Request

```
{
 "_links": {
  "authReversal": {
   "method": "POST",
   "href": "/pts/v2/payments/6482385519226028804003/reversals"
  },
  "self": {
   "method": "GET",
   "href": "/pts/v2/payments/648238551922602880 4003"
  },
  "capture": {
   "method": "POST",
   "href": "/pts/v2/payments/6482385519226028804003/captures"
  }
 },
 "clientReferenceInformation": {
  "code": "1648238551902"
 },
 "id": "6482385519226028804003",
 "orderInformation": {
  "amountDetails": {
   "authorizedAmount": "117.94",
   "currency": "USD"
  }
 },
 "paymentAccountInformation": {
  "card": {
   "type": "001"
  }
 },
 "paymentInformation": {
  "tokenizedCard": {
   "type": "001"
  },
```

```
    "card": {
      "type": "001"
    }
  },
  "processorInformation": {
    "systemTraceAuditNumber": "191521",
    "approvalCode": "831000",
    "merchantAdvice": {
      "code": "01",
      "codeRaw": "M001"
    },
    "responseDetails": "ABC",
    "networkTransactionId": "016153570198200",
    "consumerAuthenticationResponse": {
      "code": "2",
      "codeRaw": "2"
    },
    "transactionId": "016153570198200",
    "responseCode": "00",
    "avs": {
      "code": "Y",
      "codeRaw": "Y"
    }
  },
  "reconciliationId": "64823855192260288040003",
  "status": "AUTHORIZED",
  "submitTimeUtc": "2022-03-25T20:02:32Z"
}
```


Authorizations with Payment Network Tokens

This section shows you how to successfully process an authorization with payment network tokens.

> **Important**
>
> Due to mandates from the Reserve Bank of India, Indian merchants cannot store personal account numbers (PAN). Use network tokens instead. For more information on network tokens, see *Network Tokenization* in the Token Management Service Developer Guide.

Endpoint

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

Required Fields for Authorizations with Payment Network Tokens

Use these required fields for processing an authorization with payment network tokens.

Important

When using relaxed requirements for address data and the expiration date, not all fields in this list are required. It is your responsibility to determine whether your account is enabled to use this feature and which fields are required. For details about relaxed requirements, see *Relaxed Requirements for Address Data and Expiration Date in Payment Transactions* on page 264.

orderInformation.amountDetails.currency

orderInformation.amountDetails.totalAmount

orderInformation.billTo.address1

orderInformation.billTo.email

orderInformation.billTo.firstName

orderInformation.billTo.lastName

paymentinformation.tokenizedCard.cryptogram

paymentinformation.tokenizedCard.expirationMonth

paymentinformation.tokenizedCard.expirationYear

paymentInformation.tokenizedCard.transactionType

Related Information

- *API field reference guide for the REST API*

Optional Fields for Authorizations with Payment Network Tokens

You can use these optional fields to include additional information when processing an authorization with a payment network token.

clientReferenceInformation.code

consumerAuthenticationInformation.cavv — **For 3-D Secure in-app transactions for Visa and JCB, set this field to the 3-D Secure cryptogram. Otherwise, set to the network token cryptogram.**

consumerAuthenticationInformation.ucafAuthenticationData — **For Mastercard requests using 3-D Secure, set this field to the Identity Check cryptogram.**

consumerAuthenticationInformation.ucafCollectionIndicator — **For Mastercard requests using 3-D Secure, set the value to 2.**

orderInformation.amountDetails.currency

orderInformation.amountDetails.totalAmount

orderInformation.billTo.address1	
orderInformation.billTo.country	
orderInformation.billTo.email	
orderInformation.billTo.firstName	
orderInformation.billTo.lastName	
orderInformation.billTo.locality	
orderInformation.billTo.postalCode	**Required only for transactions in the US and Canada.**
orderInformation.billTo.administrativeArea	**Required only for transactions in the US and Canada.**
processingInformation.commerceIndicator	
paymentInformation.tokenizedCard.cardType	**It is strongly recommended that you send the card type even if it is optional for your processor. Omitting the card type can cause the transaction to be processed with the wrong card type.**
paymentInformation.tokenizedCard.cryptogram	
paymentInformation.tokenizedCard.expirationMonth	**Set to the token expiration month that you received from the token service provider.**
paymentInformation.tokenizedCard.expirationYear	**Set to the token expiration year that you received from the token service provider.**
paymentInformation.tokenizedCard.number	**Set to the token value that you received from the token service provider.**
paymentInformation.tokenizedCard.requestorID	**Required on Visa Platform Connect**
paymentInformation.tokenizedCard.transactionType	

Related Information

- *API field reference guide for the REST API*

REST Example: Authorizations with Payment Network Tokens

Endpoint:
Production: POST https://api.cybersource.com/pts/v2/payments
Test: POST https://apitest.cybersource.com/pts/v2/payments

Request

```
{
  "orderInformation" : {
    "amountDetails" : {
      "totalAmount" : "100",
```

```
    "currency" : "USD"
   }
  },
   "paymentInformation" : {
   "tokenizedCard" : {
    "expirationYear" : "2031",
    "number" : "4111111111111111",
    "expirationMonth" : "12",
    "transactionType" : "1",
    "cryptogram" : "qE5juRwDzAUFBAkEHuWW9PiBkWv="
   }
  }
}
```

Response to a Successful Request

```
{
  "_links": {
    "authReversal": {
      "method": "POST",
      "href": "/pts/v2/payments/6838294805206235603954/reversals"
    },
    "self": {
      "method": "GET",
      "href": "/pts/v2/payments/683829480520623560 3954"
    },
    "capture": {
      "method": "POST",
      "href": "/pts/v2/payments/6838294805206235603954/captures"
    }
  },
  "clientReferenceInformation": {
    "code": "1683829480593"
  },
  "id": "6838294805206235603954",
  "orderInformation": {
    "amountDetails": {
      "authorizedAmount": "100.00",
      "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": "1"
      }
    },
    "reconciliationId": "60332034UHI9PRJ0",
    "status": "AUTHORIZED",
    "submitTimeUtc": "2023-05-11T18:24:40Z"
}
```


Authorizations with a Card Verification Number

This section shows you how to process an authorization with a Card Verification Number (CVN).

CVN Results

The response includes a raw response code and a mapped response code:

- The raw response code is the value returned by the processor. This value is returned in the **processorInformation.cardVerification.resultCodeRaw** field. Use this value only for debugging purposes; do not use it to determine the card verification response.
- The mapped response code is the pre-defined value that corresponds to the raw response code. This value is returned in the **processorInformation.cardVerification.resultCode** field.

Even when the CVN does not match the expected value, the issuing bank might still authorize the transaction. You will receive a CVN decline, but you can still capture the transaction because it has been authorized by the bank. However, you must review the order to ensure that it is legitimate.

Settling authorizations that fail the CVN check might have an impact on the fees charged by your bank. Contact your bank for details about how card verification management might affect your discount rate.

When a CVN decline is received for the authorization in a sale request, the capture request is not processed unless you set the **processingInformation.authorizationOptions.ignoreCvResult** field to `true`.

CVN Results for American Express	A value of `1` in the **processorInformation.cardVerification.resultCode** field indicates that your account is not configured to use card verification. Contact

	customer support to have your account enabled for this feature.
CVN Results for Discover	When the CVN does not match, Discover refuses the card and the request is declined. The reply message does not include the **processorInformation.cardVerification.resultCode** field, which indicates that the CVN failed.
CVN Results for Visa and Mastercard	A CVN code of `D` or `N` causes the request to be declined with a reason code value of `230`. You can still capture the transaction, but you must review the order to ensure that it is legitimate. Cybersource, not the issuer, assigns the CVN decline to the authorization. You can capture any authorization that has a valid authorization code from the issuer, even when the request receives a CVN decline. When the issuer does not authorize the transaction and the CVN does not match, the request is declined because the card is refused. You cannot capture the transaction.

Fields Specific to this Use Case

Include this field with a standard authorization request when processing an authorization with a CVN:

- **paymentInformation.card.securityCode**

Endpoint

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

Required Fields for Processing an Authorization with a Card Verification Number

Use these required fields for processing an authorization that includes a Card Verification Number (CVN).

Important

When using relaxed requirements for address data and the expiration date, not all fields in this list are required. It is your responsibility to determine whether your account is enabled to use this feature and which fields are required. For details

about relaxed requirements, see *Relaxed Requirements for Address Data and Expiration Date in Payment Transactions* on page 264.

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.securityCode

paymentInformation.card.type

paymentInformation.card.securityCode

Related Information

- *API field reference guide for the REST API*

Optional Fields for Processing an Authorization with a Card Verification Number

You can use these optional fields to include additional information when processing an authorization with a card verification number.

paymentInformation.card.securityCodeIndicator

processingInformation.authorizationOptions.ignoreCvResult

REST Example: Processing an Authorization with a Card Verification Number

Request

```
{
  "paymentInformation": {
    "card": {
```

```
      "number": "4111111111111111",
      "expirationMonth": "12",
      "expirationYear": "2031",
      "type": "001",
      "securityCode": "999"
      }
   },
   "orderInformation": {
      "amountDetails": {
      "totalAmount": "49.95",
      "currency": "USD"
   },
    "billTo": {
      "firstName": "John",
      "lastName": "Doe",
      "address1": "1295 Charleston Rd.",
      "locality": "Mountain View",
      "administrativeArea": "CA",
      "postalCode": "94043",
      "country": "US",
      "email": "jdoe@example.com",
      "phoneNumber": "650-965-6000"
      }
   }
}
```

Response to a Successful Request

```
{
   "_links": {
      "authReversal": {
         "method": "POST",
         "href": "/pts/v2/payments/6554147587216874903954/reversals"
      },
      "self": {
         "method": "GET",
         "href": "/pts/v2/payments/655414758721687 4903954"
      },
      "capture": {
         "method": "POST",
         "href": "/pts/v2/payments/6554147587216874903954/captures"
      }
   },
   "clientReferenceInformation": {
      "code": "1655414758839"
   },
   "id": "6554147587216874903954",
   "orderInformation": {
      "amountDetails": {
         "authorizedAmount": "49.95",
         "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": "67546603C43Z6JWN",
    "status": "AUTHORIZED",
    "submitTimeUtc": "2022-06-16T21:25:58Z"
}
```


Authorizations with Foreign Merchants

Visa mandates marketplaces identify domestic marketplace transactions where the marketplace and issuer are in the same country, but the retailer is in a different country. For marketplaces in the European Economic Area (EEA) and the UK (and Gibraltar), this includes transactions where the marketplace and the issuer are within the EEA, UK, and Gibraltar, but the retailer is not located within the EEA, UK, and Gibraltar. This is flagged using the Foreign Retail Indicator (FRI).

Important

This feature is intended to be used during a settlement. While you can include this information in an authorization, this is not the preferred method.

When you include the **merchantInformation.merchantDescriptor.country** and **aggregatorInformation.subMerchant.country** fields and the merchant and submerchant are located in separate locations, within the authorization request, the transaction includes the foreign retail indicator flag. However, this flag is only submitted during settlement process and if the same fields are included in the capture request, the capture request will override the information included in the authorization request.

Fields Specific to this Use Case

These fields are required for this use case:

merchantInformation.merchantDescriptor.country	If the Merchant country is not submitted, the merchant country associated with the transaction MID will be used.
aggregatorInformation.subMerchant.country	If the Merchant country is not submitted, the merchant country associated with the transaction MID will be used.

Endpoint

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

Required Fields for Processing a Basic Authorization

Use these required fields for processing a basic authorization.

> **Important**
>
> When using relaxed requirements for address data and the expiration date, not all fields in this list are required. It is your responsibility to determine whether your account is enabled to use this feature and which fields are required. For details about relaxed requirements, see *Relaxed Requirements for Address Data and Expiration Date in Payment Transactions* on page 264.

aggregatorInformation.subMerchant.country

merchantInformation.merchantDescriptor.country

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

Related Information

- *API field reference guide for the REST API*

REST Example: Processing an Authorization with a Foreign Merchant

Endpoint:

- Production: POST https://api.cybersource.com/pts/v2/payments
- Test: POST https://apitest.cybersource.com/pts/v2/payments

Request

```
{
  "aggregatorInformation" : {
    "subMerchant" : {
      "country" : "AU"
    }
  },
  {
  "orderInformation": {
    "billTo": {
      "country": "US",
      "lastName": "Kim",
      "address1": "201 S. Division St.",
      "postalCode": "48104-2201",
      "locality": "Ann Arbor",
      "administrativeArea": "MI",
      "firstName": "Kyong-Jin",
      "email": "test@cybs.com"
    },
    "amountDetails": {
      "totalAmount": "100.00",
      "currency": "usd"
    }
  },
  {
  "merchantInformation" : {
    "merchantDescriptor" : {
      "country" : "GB"
    }
  },
  {
  "paymentInformation": {
    "card": {
      "expirationYear": "2031",
      "number": "4111111111111111",
      "expirationMonth": "12",
      "type": "001"
    }
  }
}
```

Response to a Successful Request


```
{
 "_links" : {
  "authReversal" : {
   "method" : "POST",
   "href" : "/pts/v2/payments/6461731521426399003473/reversals"
  },
  "self" : {
   "method" : "GET",
   "href" : "/pts/v2/payments/646173152142639900 3473"
  },
  "capture" : {
   "method" : "POST",
   "href" : "/pts/v2/payments/6461731521426399003473/captures"
  }
 },
 "clientReferenceInformation" : {
  "code" : "1646173152047"
 },
 "id" : "6461731521426399003473",
 "orderInformation" : {
  "amountDetails" : {
   "authorizedAmount" : "100.00",
   "currency" : "usd"
  }
 },
 "paymentAccountInformation" : {
  "card" : {
   "type" : "001"
  }
 },
 "paymentInformation" : {
  "tokenizedCard" : {
   "type" : "001"
  },
  "card" : {
   "type" : "001"
  }
 },
"paymentInsightsInformation" : {
  "responseInsights" : {
   "categoryCode" : "01"
  }
 },
 "processorInformation" : {
  "systemTraceAuditNumber" : "862481",
  "approvalCode" : "831000",
  "merchantAdvice" : {
   "code" : "01",
   "codeRaw" : "M001"
  },
  "responseDetails" : "ABC",
  "networkTransactionId" : "016153570198200",
  "consumerAuthenticationResponse" : {
   "code" : "2",
```

```
    "codeRaw" : "2"
   },
   "transactionId" : "016153570198200",
   "responseCode" : "00",
   "avs" : {
    "code" : "Y",
    "codeRaw" : "Y"
   }
  },
  "reconciliationId" : "6461731521426399003473",
  "status" : "AUTHORIZED",
  "submitTimeUtc" : "2022-03-01T22:19:12Z"
}
```

Response to a Declined Request

```
{
  "clientReferenceInformation": {
    "code": "TC50171_3"
  },
  "errorInformation": {
    "reason": "PROCESSOR_ERROR",
    "message": "Invalid account"
  },
  "id": "6583553837826789303954",
  "paymentInsightsInformation": {
    "responseInsights": {
      "categoryCode": "01",
      "category": "ISSUER_WILL_NEVER_APPROVE"
    }
  },
  "pointOfSaleInformation": {
    "amexCapnData": "1009S0600100"
  },
  "processorInformation": {
    "systemTraceAuditNumber": "004544",
    "merchantNumber": "1231231222",
    "networkTransactionId": "431736869536459",
    "transactionId": "431736869536459",
   "responseCode": "111",
    "avs": {
      "code": "Y",
      "codeRaw": "Y"
    }
  },
  "status": "DECLINED"
}
```

Authorizations with Strong Customer Authentication Exemption

This section shows you how to process an authorization with a strong customer authentication (SCA) exemption.
You can use SCA exemptions to streamline the payment process. SCA exemptions are part of the European second Payment Services Directive (PSD2) and allow certain types of low-risk transactions to bypass additional authentication steps while still remaining compliant with PSD2. You can choose which exemption can be applied to a transaction, but the card-issuing bank actually grants an SCA exemption during card authentication.
You can process an authorization with two types of SCA exemptions:

- Exemption on Authorization: Send an authorization without payer authentication and request an SCA exemption on the authorization. If it is not approved, you may be required to request further authentication upon retry.
- Exemption on Authentication: Request an SCA exemption during payer authentication and if successful, send an authorization including the SCA exemption details.

Depending on your processor, use one of these exemption fields:

Important

If you send more than one SCA exemption field with a single authentication, the transaction is denied.

- Authentication Outage: Payer authentication is not available for this transaction due to a system outage.
- B2B Corporate Card: Payment cards specifically for business-to-business transactions are exempt.
- Delegated Authentication: Payer authentication was performed outside of the authorization workflow.
- Follow-On Installment Payment: Installment payments of a fixed amount are exempt after the first transaction.
- Follow-On Recurring Payment: Recurring payments of a fixed amount are exempt after the first transaction.
- Low Risk: The average fraud levels associated with this transaction are considered low.
- Low Value: The transaction value does not warrant SCA.
- Merchant Initiated Transactions: As follow-on transactions, merchant-initiated transactions are exempt.
- Stored Credential Transaction: Credentials are authenticated before storing, so stored credential transactions are exempt.
- Trusted Merchant: Merchants registered as trusted beneficiaries.

Exemption Fields Specific to the Strong Customer Authentication Use Case

Use one of these fields to request an SCA exemption:

Types of SCA Exemptions

Exemption Type	Field	Value
Authentication Outage	consumerAuthenticationInformation. strongAuthentication. authenticationOutageExemptionIndicator	1
B2B Corporate Card Transaction	consumerAuthenticationInformation. strongAuthentication. secureCorporatePaymentIndicator	1
Delegated Authentication	consumerAuthenticationInformation. strongAuthentication. delegatedAuthentication ExemptionIndicator	1
Low-Risk Transaction	consumerAuthenticationInformation. strongAuthentication. riskAnalysisExemptionIndicator	1
Low-Value Transaction	consumerAuthenticationInformation. strongAuthentication. lowValueExemptionIndicator	1
Trusted Merchant Transaction	consumerAuthenticationInformation. strongAuthentication. trustedMerchantExemptionIndicator	1

Country-Specific Requirements

These fields are specific to certain countries and regions.

Argentina

merchantInformation.taxId	Required for Mastercard transactions.
merchantInformation.transactionLocalDateTime	Required when the time zone is not included in your account. Otherwise, this field is optional.

Brazil

paymentInformation.card.sourceAccountType	Required for combo card transactions.
paymentInformation.card.sourceAccountTypeDetails	Required for combo card line-of-credit and prepaid-card transactions.

Chile

merchantInformation.taxId	Required for Mastercard transactions.

Paraguay

merchantInformation.taxId Required for Mastercard transactions.

Saudi Arabia

processingInformation.authorizationOptions.transactionMode

Taiwan

paymentInformation.card.hashedNumber

Endpoint

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

Required Fields for Processing an Authorization with an SCA Exemption

Use these required fields for processing an authorization that includes an SCA exemption.

Important

When using relaxed requirements for address data and the expiration date, not all fields in this list are required. It is your responsibility to determine whether your account is enabled to use this feature and which fields are required. For details about relaxed requirements, see *Relaxed Requirements for Address Data and Expiration Date in Payment Transactions* on page 264.

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: Processing an Authorization with an SCA Exemption for Low-Value Transactions

Endpoint:

- Production: POST https://api.cybersource.com/pts/v2/payments
- Test: POST https://apitest.cybersource.com/pts/v2/payments

Request

```
{
 "consumerAutenticationInformation" : {
  "strongAuthentication" : {
   "lowValueExemptionIndicator" : "1"
  }
 },
 "orderInformation" : {
  "billTo" : {
   "country" : "US",
   "lastName" : "Kim",
   "address1" : "201 S. Division St.",
   "postalCode" : "48104-2201",
   "locality" : "Ann Arbor",
   "administrativeArea" : "MI",
   "firstName" : "Kyong-Jin",
   "email" : "test@cybs.com"
  },
  "amountDetails" : {
   "totalAmount" : "100.00",
   "currency" : "eur"
  }
 },
 "paymentInformation" : {
  "card" : {
   "expirationYear" : "2031",
   "number" : "4111111111111111",
   "expirationMonth" : "12"
  }
 }
}
```

Response to a Successful Request

```
{
   "_links": {
      "authReversal": {
         "method": "POST",
         "href": "/pts/v2/payments/6709780221406171803955/reversals"
      },
      "self": {
         "method": "GET",
```

```
        "href": "/pts/v2/payments/6709780221406171803955"
      },
      "capture": {
        "method": "POST",
        "href": "/pts/v2/payments/6709780221406171803955/captures"
      }
    },
    "clientReferenceInformation": {
      "code": "1670978022258"
    },
    "id": "6709780221406171803955",
    "orderInformation": {
      "amountDetails": {
        "authorizedAmount": "100.00",
        "currency": "eur"
      }
    },
    "paymentAccountInformation": {
      "card": {
        "type": "001"
      }
    },
    "paymentInformation": {
      "tokenizedCard": {
        "type": "001"
      },
      "card": {
        "type": "001"
      }
    },
    "pointOfSaleInformation": {
      "terminalId": "123456"
    },
    "processorInformation": {
      "approvalCode": "888888",
      "networkTransactionId": "123456789619999",
      "transactionId": "123456789619999",
      "responseCode": "100",
      "avs": {
        "code": "X",
        "codeRaw": "I1"
      }
    },
    "reconciliationId": "62859554PBDEMI43",
    "status": "AUTHORIZED",
    "submitTimeUtc": "2022-12-14T00:33:42Z"
}
```


Zero Amount Authorizations

Authorizing a payment for a zero amount shows whether a payment card account is valid and whether the card is lost or stolen. You cannot capture a zero amount authorization. This section provides information you need to process a zero amount authorization.

Processor-Specific Information

Visa Platform Connect	**AVS and CVN are supported.** **Supported for Internet, MOTO, and card-present transactions. Do not try to perform a zero amount authorization for a recurring payment, installment payment, or payer authorization transaction.**

Endpoint

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

Required Fields for Processing a Zero Amount Authorization

Use these required fields for processing a zero amount authorization.

Important

When using relaxed requirements for address data and the expiration date, not all fields in this list are required. It is your responsibility to determine whether your account is enabled to use this feature and which fields are required. For details about relaxed requirements, see *Relaxed Requirements for Address Data and Expiration Date in Payment Transactions* on page 264.

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

Related Information

- *API field reference guide for the REST API*

Country-Specific Required Fields for Processing a Zero Amount Authorization

Use these country-specific required fields to process a zero amount authorization.

Argentina

merchantInformation.taxId	**Required for Mastercard transactions.**
merchantInformation.transactionLocalDateTime	**Required in Argentina when the time zone is not included in your account. Otherwise, this field is optional.**

Brazil

paymentInformation.card.sourceAccountType	**Required for combo card transactions.**
paymentInformation.card.sourceAccountTypeRequired	**Required for combo card line-of-credit and prepaid-card transactions.**

Saudi Arabia

processingInformation.authorizationOptions.transactionMode

Taiwan

paymentInformation.card.hashedNumber

Related Information

- *API field reference guide for the REST API*

REST Example: Processing a Zero Amount Authorization

Endpoint:

- Production: POST https://api.cybersource.com/pts/v2/payments
- Test: POST https://apitest.cybersource.com/pts/v2/payments

Request

```
{
  "orderInformation" : {
    "billTo" : {
      "country" : "US",
      "lastName" : "Kim",
      "address1" : "201 S. Division St.",
```

```
    "postalCode" : "48104-2201",
    "locality" : "Ann Arbor",
    "administrativeArea" : "MI",
    "firstName" : "Kyong-Jin",
    "email" : "test@cybs.com"
   },
   "amountDetails" : {
    "totalAmount" : "0.00",
    "currency" : "usd"
   }
  },
  "paymentInformation" : {
   "card" : {
    "expirationYear" : "2031",
    "number" : "4111111111111111",
    "expirationMonth" : "12"
   }
  }
}
```

Response to a Successful Request

```
{
  "_links" : {
   "authReversal" : {
    "method" : "POST",
    "href" : "/pts/v2/payments/6461731521426399003473/reversals"
   },
   "self" : {
    "method" : "GET",
    "href" : "/pts/v2/payments/646173152142639900 3473"
   },
   "capture" : {
    "method" : "POST",
    "href" : "/pts/v2/payments/6461731521426399003473/captures"
   }
  },
  "clientReferenceInformation" : {
   "code" : "1646173152047"
  },
  "id" : "6461731521426399003473",
  "orderInformation" : {
   "amountDetails" : {
    "authorizedAmount" : "0",
    "currency" : "usd"
   }
  },
  "paymentAccountInformation" : {
   "card" : {
    "type" : "001"
   }
  },
  "paymentInformation" : {
   "tokenizedCard" : {
    "type" : "001"
   },
```

```
    "card" : {
     "type" : "001"
    }
   },
   "processorInformation" : {
    "systemTraceAuditNumber" : "862481",
    "approvalCode" : "831000",
    "merchantAdvice" : {
     "code" : "01",
     "codeRaw" : "M001"
    },
    "responseDetails" : "ABC",
    "networkTransactionId" : "016153570198200",
    "consumerAuthenticationResponse" : {
     "code" : "2",
     "codeRaw" : "2"
    },
    "transactionId" : "016153570198200",
    "responseCode" : "00",
    "avs" : {
     "code" : "Y",
     "codeRaw" : "Y"
    }
   },
   "reconciliationId" : "6461731521426399003473",
   "status" : "AUTHORIZED",
   "submitTimeUtc" : "2022-03-01T22:19:12Z"
}
```


Incremental Authorizations

This section shows you how to process an incremental authorization.
Incremental authorizations allow merchants to add additional products and services to an existing authorization. This section will show you how to append an original authorization to include additional transactions.
The supported card types for incremental authorizations are Mastercard and Visa.
The incremental authorization service has these limitations:

- Maximum of 100 incremental authorizations per transaction, in addition to the initial authorization.
- Interchange optimization is not supported.
- Split shipments are not supported.

Endpoint

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

Required Fields for Processing an Incremental Authorization

Use these required fields for processing an incremental authorization.

Important

When using relaxed requirements for address data and the expiration date, not all fields in this list are required. It is your responsibility to determine whether your account is enabled to use this feature and which fields are required. For details about relaxed requirements, see *Relaxed Requirements for Address Data and Expiration Date in Payment Transactions* on page 264.

clientReferenceInformation.code	
id	**Set the id URL parameter to the request ID that was included in the original authorization response message.**
orderInformation.amountDetails.currency	
orderInformation.amountDetails.totalAmount	
processingInformation.authorizationOptions.initiator.merchantInitiatedTransaction.reason	**Set the value to 5.**

Related Information

- *API field reference guide for the REST API*

Country-Specific Required Fields for Processing an Incremental Authorization

Use these country-specific required fields to process an incremental authorization.

Argentina

merchantInformation.transactionLocalDateTime	**Required in Argentina when the time zone is not included in your account. Otherwise, this field is optional.**

Related Information

- *API field reference guide for the REST API*

Optional Field for Processing an Incremental Authorization

You can use this optional field to include additional information when processing an incremental authorization.

travelInformation.duration

Related Information

- *API field reference guide for the REST API*

REST Example: Processing an Incremental Authorization

Endpoint:

- Production: POST https://api.cybersource.com/pts/v2/payments
- Test: POST https://apitest.cybersource.com/pts/v2/payments

Request

```
{
"processingInformation": {
   "authorizationOptions": {
    "initiator": {
     "storedCredentialUsed": "true"
    },
    "merchantInitiatedTransaction": {
     "reason": "5"
    }
   }
 },
 "orderInformation" : {
  "billTo" : {
   "country" : "US",
   "lastName" : "Singh",
   "address1" : "201 S. Division St.",
   "postalCode" : "48104-2201",
   "locality" : "Ann Arbor",
   "administrativeArea" : "MI",
   "firstName" : "Ranjeeth",
   "district" : "MI",
   "email" : "test@cybs.com"
  },
  "amountDetails" : {
   "additionalAmount": "101.00",
   "totalAmount" : "100.00",
   "currency" : "usd"
  }
 },
 "paymentInformation" : {
  "card" : {
   "expirationYear" : "2031",
   "number" : "4111111111111111",
   "securityCode" : "123",
   "expirationMonth" : "12",
   "type" : "001"
  }
 },
"merchantInformation": {
   "transactionLocalDateTime": "20191002080000"
 },
 "id": "3434254100000181552556"
}
```

Response to a Successful Request

```
{
```

```
  "_links" : {
   "authReversal" : {
    "method" : "POST",
    "href" : "/pts/v2/payments/6479624584536070903093/reversals"
   },
   "self" : {
    "method" : "GET",
    "href" : "/pts/v2/payments/647962458453607090393"
   },
   "capture" : {
    "method" : "POST",
    "href" : "/pts/v2/payments/6479624584536070903093/captures"
   }
  },
  "clientReferenceInformation" : {
   "code" : "33557799"
  },
  "id" : "647962458453607090393",
  "orderInformation" : {
   "amountDetails" : {
    "authorizedAmount" : "100.00",
    "currency" : "usd"
   }
  },
  "paymentAccountInformation" : {
   "card" : {
    "type" : "001"
   }
  },
  "paymentInformation" : {
   "tokenizedCard" : {
    "type" : "001"
   },
   "card" : {
    "type" : "001"
   }
  },
  "processorInformation" : {
   "systemTraceAuditNumber" : "819203",
   "approvalCode" : "831000",
   "cardVerification" : {
    "resultCodeRaw" : "M",
    "resultCode" : "M"
   },
   "merchantAdvice" : {
    "code" : "01",
    "codeRaw" : "M001"
   },
   "responseDetails" : "ABC",
   "networkTransactionId" : "016153570198200",
   "retrievalReferenceNumber" : "208115819203",
   "consumerAuthenticationResponse" : {
    "code" : "2",
    "codeRaw" : "2"
   },
   "transactionId" : "016153570198200",
```

```
    "responseCode" : "00",
    "avs" : {
      "code" : "Y",
      "codeRaw" : "Y"
    }
  },
  "reconciliationId" : "647962458453607090309 3",
  "status" : "AUTHORIZED",
  "submitTimeUtc" : "2022-03-22T15:20:58Z"
}
```


Final Authorization Indicator

The purpose of this feature is to ensure that unused funds are reversed, so that customer's funds are available again when an order is not fulfilled.
For an authorization with an amount greater than zero, indicate whether the authorization is a final authorization, a preauthorization, or an undefined authorization.
You can set a default authorization type in your account. To set the default authorization type in your account, contact customer support.
Chargeback protection is in effect for seven days after the authorization.

Supported Services

- Authorization
- Incremental authorization

Supported Card Types

- Co-badged Mastercard and mada. You must identify the card type as Mastercard. Supported only on Visa Platform Connect.
- Maestro (International)
- Maestro (UK Domestic)
- Mastercard

Requirements for Final Authorizations

For a final authorization:

- The authorization amount must be greater than zero.
- The authorization amount must be the final amount that the customer agrees to pay.
- The authorization should not be cancelled after it is approved except when a system failure occurs.
- The authorization must be submitted for capture within seven calendar days of its request.
- The capture amount and currency must be the same as the authorization amount and currency.

Pre-authorizations

A pre-authorization enables merchants to authorize a charge when the final amount is unknown, which is typical for hotel, auto rental, e-commerce, and restaurant transactions. For a pre-authorization:

- The authorization amount must be greater than zero.
- The authorization must be submitted for capture within 30 calendar days of its request.
- When you do not capture the authorization, you must reverse it.
  In the U.S., Canada, Latin America, and Asia Pacific, Mastercard charges an additional fee for a preauthorization that is not captured and not reversed.
  In Europe, Russia, Middle East, and Africa, Mastercard charges fees for all pre-authorizations.
- Chargeback protection is in effect for 30 days after the authorization.

Unmarked Authorizations

An authorization is unmarked when the default authorization type is not set in your account and you do not include the **authIndicator** field in the authorization request. To set the default authorization type in your account, contact customer support.
Unmarked authorizations are supported only in the US, Canada, Latin America, and Asia Pacific. They are not supported in Europe, Russia, Middle East, and Africa.
Cybersource does not set a mark or indicator for the type of authorization in the request that is sent to the processor.

Important

Your acquirer processes an unmarked authorization as a final authorization, a preauthorization, or an undefined authorization. Contact your acquirer to learn how they process unmarked authorizations.

Requirements for Unmarked Authorizations

For an unmarked authorization:

- The authorization amount must be greater than zero.
- The authorization amount can be different from the final transaction amount.

Undefined Authorizations

An authorization is undefined when you set the default authorization type in your account to undefined and do not include the **authIndicator** field in the authorization request. To set the default authorization type in your account, contact customer support.
Undefined authorizations are supported only in the U.S., Canada, Latin America, and Asia Pacific. They are not supported in Europe, Russia, Middle East, and Africa.
Chargeback protection is in effect for seven days after the authorization.

Requirements for Undefined Authorizations

For an undefined authorization:

- The authorization amount must be greater than zero.
- The authorization amount can be different from the final transaction amount.
- The authorization should not be cancelled after it is approved except when a system failure occurs.
- The authorization must be submitted for capture within seven calendar days of its request.
- When you do not capture the authorization, you must reverse it; otherwise, Mastercard charges an additional fee for the transaction.

Required Fields for Final Authorizations

Use these required fields for final authorizations and preauthorizations.

Important

When using relaxed requirements for address data and the expiration date, not all fields in this list are required. It is your responsibility to determine whether your account is enabled to use this feature and which fields are required. For details about relaxed requirements, see *Relaxed Requirements for Address Data and Expiration Date in Payment Transactions* on page 264.

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.type

paymentInformation.card.expirationMonth

paymentInformation.card.expirationYear

paymentInformation.card.number

processingInformation.authorizationOptions.settlementIndicator

Set the value to `0` for preauthorizations, or to `1` for final authorizations. Do not include this field for unmarked or undefined authorizations.

REST Example: Final Authorizations

Endpoint:

- Production: https://api.cybersource.com/pts/v2/payments
- Test: POST https://apitest.cybersource.com/pts/v2/payments

Request

```
{
 "orderInformation" : {
  "billTo" : {
   "firstName" : "RTS",
   "lastName" : "VDP",
   "address1" : "201 S. Division St.",
   "postalCode" : "48104-2201",
   "locality" : "Ann Arbor",
   "administrativeArea" : "MI",
   "country" : "US",
   "email" : "test@cybs.com"
  },
  "amountDetails" : {
   "totalAmount" : "100.00",
   "currency" : "usd"
  }
 },
 "paymentInformation" : {
  "card" : {
   "expirationYear" : "2031",
   "number" : "4111111111111111",
   "expirationMonth" : "12",
   "type" : "001"
  }
 },
 "processingInformation" : {
  "authorizationOptions" : {
    "authIndicator" : "1"
  }
 }
}
```

Response to a Successful Request

```
{
  "_links": {
    "authReversal": {
      "method": "POST",
      "href": "/pts/v2/payments/6910040807416719003955/reversals"
    },
    "self": {
      "method": "GET",
      "href": "/pts/v2/payments/6910040807416719003955"
    },
    "capture": {
      "method": "POST",
```

```
        "href": "/pts/v2/payments/6910040807416719003955/captures"
      }
    },
    "clientReferenceInformation": {
      "code": "1691004080800"
    },
    "id": "6910040807416719003955",
    "orderInformation": {
      "amountDetails": {
        "authorizedAmount": "100.00",
        "currency": "usd"
      }
    },
    "paymentAccountInformation": {
      "card": {
        "type": "001"
      }
    },
    "paymentInformation": {
      "card": {
        "type": "001"
      }
    },
    "pointOfSaleInformation": {
      "terminalId": "111111"
    },
    "processorInformation": {
      "approvalCode": "888888",
      "networkTransactionId": "123456789619999",
      "transactionId": "123456789619999",
      "responseCode": "100",
      "avs": {
        "code": "X",
        "codeRaw": "I1"
      }
    },
    "reconciliationId": "67628631TKRG2OVE",
    "status": "AUTHORIZED",
    "submitTimeUtc": "2023-08-02T19:21:20Z"
}
```


Authorization Reversals

This section provides the information you need in order to process an authorization reversal.
Reversing an authorization releases the hold on the customer's payment card funds that the issuing bank placed when processing the authorization.

Endpoint

Production: POST https://api.cybersource.com/pts/v2/payments/{id}/reversals
Test: POST https://apitest.cybersource.com/pts/v2/payments/{id}/reversals

The {id} is the transaction ID returned in the authorization response.

Required Fields for Processing an Authorization Reversal

clientReferenceInformation.code	
clientReferenceInformation.partner.thirdPartyCertificationNumber	**Cybersource provides the value for this field.**
id	**Set the id URL parameter to the transaction ID that was included in the authorization response message.**
orderInformation.amountDetails.currency	
reversalInformation.amountDetails.totalAmount	**The amount of the reversal must be the same as the authorization amount that was included in the authorization response message. Do not use the amount that was requested in the authorization request message.**

REST Example: Processing an Authorization Reversal

Endpoint

Production: POST https://api.cybersource.com/pts/v2/payments/{id}/reversals
Test: POST https://apitest.cybersource.com/pts/v2/payments/{id}/reversals
For this example, the {id} portion of the URL is set to the transaction ID included in the authorization you want to void: 686945868586643800 03955

Request

```
{
  "clientReferenceInformation": {
   "code": "test123",
   "partner": {
     "thirdPartyCertificationNumber": "testTPCN"
   }
  }
  "orderInformation" : {
    "amountDetails" : {
      "currency" : "USD"
    }
  },
  "reversalInformation" : {
    "amountDetails" : {
      "totalAmount" : "100.00"
    }
  }
}
```

Response to a Successful Request


```
{
  "_links" : {
   "self" : {
     "method" : "GET",
     "href" : "/pts/v2/reversals/6869460219566537303955"
   }
  },
  "clientReferenceInformation" : {
    "code" : "RTS-Auth-Reversal"
  },
  "id" : "6869460219566537303955",
  "orderInformation" : {
    "amountDetails" : {
      "currency" : "USD"
    }
  },
  "processorInformation" : {
    "responseCode" : "200"
  },
  "reconciliationId" : "82kBK3qDNtls",
  "reversalAmountDetails" : {
    "reversedAmount" : "100.00",
    "currency" : "USD"
  },
  "status" : "REVERSED",
  "submitTimeUtc" : "2023-06-16T20:07:02Z"
}
```


Timeout Authorization Reversals

When you do not receive a response message after sending an authorization request, this feature enables you to reverse the authorization that you requested.

Important

Wait 60 seconds before requesting a timeout authorization reversal.

Include the **clientReferenceInformation.transactionId** field in the original request for an authorization. The value of the merchant transaction ID must be unique for 180 days. When the original transaction fails, the response message for the reversal request includes these fields:

- **reversalAmountDetails.originalTransactionAmount**
- **processorInformation.responseCode**

Requirements

Unless your processor supports authorization reversal after void (ARAV), timeout authorization reversals are supported only for authorizations that have not been captured and settled.

Endpoint

Production: POST https://api.cybersource.com/pts/v2/reversals
Test: POST https://apitest.cybersource.com/pts/v2/reversals

Required Fields for Processing a Timeout Authorization Reversal

Use these required fields for processing a timeout authorization reversal.

clientReferenceInformation.transactionId	**Identifier that links the reversal request to the original request.**
reversalInformation.amountDetails.currency	
reversalInformation.amountDetails.totalAmount	**The amount of the reversal must be the same as the authorization amount that was included in the authorization response message. Do not use the amount that was requested in the authorization request message.**

Related Information

- *REST API Field Reference*

REST Example: Processing a Timeout Authorization Reversal

Endpoint

Production: POST https://api.cybersource.com/pts/v2/reversals
Test: POST https://apitest.cybersource.com/pts/v2/reversals

Request

```
{
 "clientReferenceInformation": {
  "transactionId": "987654321"
 },
  "reversalInformation": {
   "amountDetails": {
    "totalAmount": "100.00",
    "currency": "USD"
   },
   "reason": "testing"
 }
}
```

Response to a Successful Request


```
{
 "_links" : {
  "self" : {
   "method" : "GET",
   "href" : "/pts/v2/reversals/6869460219566537303955"
  }
 },
 "clientReferenceInformation" : {
  "code" : "RTS-Auth-Reversal"
 },
 "id" : "6869460219566537303955",
 "orderInformation" : {
  "amountDetails" : {
   "currency" : "USD"
  }
 },
 "processorInformation" : {
  "responseCode" : "200"
 },
 "reconciliationId" : "82kBK3qDNtls",
 "reversalAmountDetails" : {
  "reversedAmount" : "100.00",
  "currency" : "USD"
 },
 "status" : "REVERSED",
 "submitTimeUtc" : "2023-06-16T20:07:02Z"
}=
```


Captures

This section provides the information you need in order to capture an authorized transaction.

Endpoint

Production: `POST https://api.cybersource.com/pts/v2/payments/{id}/captures`
Test: `POST https://apitest.cybersource.com/pts/v2/payments/{id}/captures`
The `{id}` is the transaction ID returned in the authorization response.

Required Fields for Capturing an Authorization

Use these required fields for capturing an authorization.

clientReferenceInformation.code	**This field value maps from the original authorization, sale, or credit transaction.**
clientReferenceInformation.partner.thirdPartyCertificationNumber	**Cybersource provides the value for this field.**

orderInformation.amountDetails.currency

orderInformation.amountDetails.totalAmount

REST Example: Capturing an Authorization

Endpoint:

- Production: POST https://api.cybersource.com/pts/v2/payments/{id}/captures
- Test: POST https://apitest.cybersource.com/pts/v2/payments/{id}/captures

Request

```
{
  "clientReferenceInformation": {
    "code": "ABC123",
    "partner": {
      "thirdPartyCertificationNumber": "123456789012"
    }
  },
  "orderInformation": {
    "amountDetails": {
      "totalAmount": "100.00",
      "currency": "EUR"
  }
}
```

Response to a Successful Request

```
{
  "_links": {
    "void": {
      "method": "POST",
      "href": "/pts/v2/captures/6662994431376681303954/voids"
    },
    "self": {
      "method": "GET",
      "href": "/pts/v2/captures/666299443137668130 3954"
    }
  },
  "clientReferenceInformation": {
    "code": "1666299443215"
  },
  "id": "6662994431376681303954",
  "orderInformation": {
    "amountDetails": {
      "totalAmount": "100.00",
      "currency": "EUR"
    }
  },
  "reconciliationId": "66535942B9CGT52U",
  "status": "PENDING",
  "submitTimeUtc": "2022-10-20T20:57:23Z"
}
```

Captures with Foreign Merchants

Visa mandates marketplaces identify domestic marketplace transactions where the marketplace and issuer are in the same country, but the retailer is in a different country. For marketplaces in the European Economic Area (EEA) and the UK (and Gibraltar), this includes transactions where the marketplace and the issuer are within the EEA, UK, and Gibraltar, but the retailer is not located within the EEA, UK, and Gibraltar. This is flagged using the Foreign Retail Indicator (FRI).
When you include the **merchantInformation.merchantDescriptor.country** and **aggregatorInformation.subMerchant.country** fields and the merchant and submerchant are located in separate locations, within the capture request, the transaction includes the foreign retail indicator flag.
If you include the **merchantInformation.merchantDescriptor.country** and **aggregatorInformation.subMerchant.country** fields in both an authorization request and a capture request, the information set in the capture request will override the information in the authorization request.

Endpoint

Production: `POST https://api.cybersource.com/pts/v2/payments/{id}/captures`
Test: `POST https://apitest.cybersource.com/pts/v2/payments/{id}/captures`
The `{id}` is the transaction ID returned in the authorization response.

Required Fields for Capturing an Authorization

Use these required fields for capturing an authorization.

clientReferenceInformation.code	**This field value maps from the original authorization, sale, or credit transaction.**
clientReferenceInformation.partner.thirdPartyCertificationNumber	**Cybersource provides the value for this field.**
orderInformation.amountDetails.currency	
orderInformation.amountDetails.totalAmount	

REST Example: Capturing an Authorization with a Foreign Merchant

Endpoint:

- Production: `POST https://api.cybersource.com/pts/v2/payments/{id}/captures`
- Test: `POST https://apitest.cybersource.com/pts/v2/payments/{id}/captures`

Request

```
{
  "aggregatorInformation" : {
    "subMerchant" : {
      "country" : "AU"
```

```
      }
   },{
   "clientReferenceInformation": {
      "code": "ABC123",
      "partner": {
         "thirdPartyCertificationNumber": "123456789012"
      }
   {
   "merchantInformation" : {
      "merchantDescriptor" : {
         "country" : "GB"
      }
   },   },
   "orderInformation": {
      "amountDetails": {
         "totalAmount": "100.00",
         "currency": "EUR"
   }
}
```

Response to a Successful Request

```
{
   "_links": {
      "void": {
         "method": "POST",
         "href": "/pts/v2/captures/6662994431376681303954/voids"
      },
      "self": {
         "method": "GET",
         "href": "/pts/v2/captures/666299443137668130 3954"
      }
   },
   "clientReferenceInformation": {
      "code": "1666299443215"
   },
   "id": "6662994431376681303954",
   "orderInformation": {
      "amountDetails": {
         "totalAmount": "100.00",
         "currency": "EUR"
      }
   },
   "reconciliationId": "66535942B9CGT52U",
   "status": "PENDING",
   "submitTimeUtc": "2022-10-20T20:57:23Z"
}
```

Multiple Partial Captures

This section shows you how to process multiple partial captures for an authorization.

This feature enables you to request multiple partial captures for one authorization. A multiple partial capture allows you to incrementally settle authorizations over time. Ensure that the total amount of all the captures does not exceed the authorized amount.

Fields Specific to This Use Case

These API request fields and values are specific to this use case:

processingInformation.captureOptions.captureSequenceNumber

processingInformation.captureOptions.totalCaptureCount

Prerequisite

Contact customer support to have your account enabled for this feature.

Limitations

Your account can be enabled for multiple partial captures or split shipments; it cannot be enabled for both features.

Endpoint

Production: `POST https://api.cybersource.com/pts/v2/payments/{id}/captures`
Test: `POST https://apitest.cybersource.com/pts/v2/payments/{id}/captures`
The `{id}` is the transaction ID returned in the authorization response.

Required Fields for Processing Multiple Partial Captures

clientReferenceInformation.code	**Set to clientReferenceInformation.code value used in corresponding authorization request.**
clientReferenceInformation.partner.thirdPartyCertificationNumber	**Cybersource provides the value for this field.**
orderInformation.amountDetails.currency	
orderInformation.amountDetails.totalAmount	
processingInformation.captureOptions.captureSequenceNumber	**For the final capture request, set this field and processingInformation.captureOptions.totalCaptureCo to the same value.** **Required if you are in the U.S.**
processingInformation.captureOptions.totalCaptureCount	**When you do not know the total number of captures that you are going to request, set this field to an estimated value or `99` for all capture requests except the final one. For the final capture request, set this field and**

processingInformation.captureOptions.captureSequen
to the same value.

Required if you are in the U.S.

Related Information

- *API field reference guide for the REST API*

REST Example: Processing Multiple Partial Captures

Endpoint:

- Production: POST https://api.cybersource.com/pts/v2/payments/{id}/captures
- Test: POST https://apitest.cybersource.com/pts/v2/payments/{id}/captures

id is the transaction ID returned in the authorization response.

Request

This example includes special fields that are not required by all processors.

```
{
 {
 "clientReferenceInformation": {
  "code": "TC50171_3"
 },
 "processingInformation": {
  "captureOptions": {
   "captureSequenceNumber": "2",
   "totalCaptureCount": "3"
  }
 },
 "orderInformation": {
  "amountDetails": {
   "totalAmount": "102.21",
   "currency": "USD"
  }
 }
}
```

Response to a Successful Request

```
{
 "_links": {
  "void": {
   "method": "POST",
   "href": "/pts/v2/captures/6742496815656503003954/voids"
  },
  "self": {
   "method": "GET",
   "href": "/pts/v2/captures/6742496815656503003954"
  }
 },
 "clientReferenceInformation": {
  "code": "TC50171_3"
```

```
  },
  "id": "6742496815656503003954",
  "orderInformation": {
    "amountDetails": {
      "totalAmount": "102.21",
      "currency": "USD"
    }
  },
  "reconciliationId": "67332020GD2G1OO1",
  "status": "PENDING",
  "submitTimeUtc": "2023-01-20T21:21:21Z"
}
```


Forced Captures

This feature allows merchants to process authorizations obtained through an organization other than Cybersource. For example, a merchant might call their processor to request a manual authorization, at which point they can request a forced capture of the authorization.
A manual authorization cannot be captured for more than the original authorization amount, and the authorization expires after seven days.

Supported Acquirers

- Banco Safra
- Bank Sinarmas (Omise Ltd.)
- BC Card Co., Ltd.
- Citibank Malaysia
- CTBC Bank Ltd.
- Sumitomo Mitsui Card Co.
- Vietnam Technological and Commercial Joint-Stock Bank

Supported Services

- Authorization

Required Fields for Forced Captures

Use these required fields for processing forced captures.

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.type	
paymentInformation.card.expirationMonth	
paymentInformation.card.expirationYear	
paymentInformation.card.number	
processingInformation.authorizationOptions.authType	**Set the value to** `verbal`**.**
processingInformation.authorizationOptions.verbalAuthCode	**Set this field to the manually obtained authorization code.**

Related Information

- *API field reference guide for the REST API*

REST Example: Forced Captures

Endpoint:

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

Request

```
{
  "orderInformation": {
    "billTo" : {
     "firstName" : "RTS",
     "lastName" : "VDP",
     "address1" : "201 S. Division St.",
     "postalCode" : "48104-2201",
     "locality" : "Ann Arbor",
     "administrativeArea" : "MI",
     "country" : "US",
     "email" : "test@cybs.com"
    },
    "amountDetails": {
      "totalAmount": "100.00",
      "currency": "usd"
    }
  },
  "paymentInformation": {
    "card": {
      "expirationYear": "2031",
      "number": "4111111111111111",
```

```
        "expirationMonth": "12",
        "type": "001"
      }
    },
    "processingInformation": {
      "authorizationOptions": {
        "authType": "verbal",
        "verbalAuthCode": "ABC123"
      }
    }
}
```

Response to a Successful Request

```
{
    "_links": {
      "authReversal": {
        "method": "POST",
        "href": "/pts/v2/payments/6915126171696653403954/reversals"
      },
      "self": {
        "method": "GET",
        "href": "/pts/v2/payments/6915126171696653403954"
      },
      "capture": {
        "method": "POST",
        "href": "/pts/v2/payments/6915126171696653403954/captures"
      }
    },
    "clientReferenceInformation": {
      "code": "TC50171_3"
    },
    "id": "6915126171696653403954",
    "orderInformation": {
      "amountDetails": {
        "authorizedAmount": "102.00",
        "currency": "USD"
      }
    },
    "paymentAccountInformation": {
      "card": {
        "type": "002"
      }
    },
    "paymentInformation": {
      "card": {
        "type": "002"
      }
    },
    "pointOfSaleInformation": {
      "terminalId": "111111"
    },
    "processorInformation": {
      "approvalCode": "ABC123"
    },
```

```
  "status": "AUTHORIZED",
  "submitTimeUtc": "2023-08-08T16:36:57Z"
}
```

Refunds

This section provides the information you need in order to process a refund, which is linked to a capture or sale. You must request a refund within 180 days of the authorization.

Endpoint

Production: POST https://api.cybersource.com/pts/v2/payments/{id}/refunds
Test: POST https://apitest.cybersource.com/pts/v2/payments/{id}/refunds
The {id} is the transaction ID returned in the capture or sale response.

Required Fields for Processing a Refund

Use these required fields for processing a refund.

orderInformation.amountDetails.currency

orderInformation.amountDetails.totalAmount

Related Information

- *API field reference guide for the REST API*

REST Interactive Example: Processing a Refund

Refund a Payment

Live Console URL: *https://developer.cybersource.com/api-reference-assets/index.html#payments_refund_refund-a-payment*

REST Example: Processing a Refund

Endpoint:

- Production: POST https://api.cybersource.com/pts/v2/payments/{id}/refunds
- Test: POST https://apitest.cybersource.com/pts/v2/payments/{id}/refunds

Request

```
{
  "orderInformation": {
    "amountDetails": {
      "totalAmount": "100.00",
      "currency": "EUR"
    }
  }
```

```
}
```

Response to a Successful Request

```
{
  "_links": {
    "void": {
      "method": "POST",
      "href": "/pts/v2/credits/6699964581696622603955/voids"
    },
    "self": {
      "method": "GET",
      "href": "/pts/v2/credits/6699964581696622603955"
    }
  },
  "clientReferenceInformation": {
    "code": "1669996458298"
  },
  "creditAmountDetails": {
    "currency": "eur",
    "creditAmount": "100.00"
  },
  "id": "6699964581696622603955",
  "orderInformation": {
    "amountDetails": {
      "currency": "EUR"
    }
  },
  "paymentAccountInformation": {
    "card": {
      "type": "001"
    }
  },
  "paymentInformation": {
    "tokenizedCard": {
      "type": "001"
    },
    "card": {
      "type": "001"
    }
  },
  "processorInformation": {
    "approvalCode": "888888",
    "responseCode": "100"
  },
  "reconciliationId": "61873329OAILG3Q6",
  "status": "PENDING",
  "submitTimeUtc": "2022-12-02T15:54:18Z"
}
```

Credit

This section shows you how to process a credit, which is not linked to a capture or sale. There is no time limit for requesting a credit.

Endpoint

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

Required Fields for Processing a Credit

Use these required fields for processing a credit.

> **Important**
>
> When using relaxed requirements for address data and the expiration date, not all fields in this list are required. It is your responsibility to determine whether your account is enabled to use this feature and which fields are required. For details about relaxed requirements, see *Relaxed Requirements for Address Data and Expiration Date in Payment Transactions* on page 264.

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

REST Interactive Example: Processing a Credit

Credit

Live Console URL: *https://developer.cybersource.com/api-reference-assets/index.html#payments_credit_process-a-credit*

REST Example: Processing a Credit

Endpoint:

- Production: POST https://api.cybersource.com/pts/v2/credits/
- Test: POST https://apitest.cybersource.com/pts/v2/credits/

Request

```
{
 "orderInformation" : {
  "billTo" : {
   "country" : "US",
   "lastName" : "Kim",
   "address1" : "201 S. Division St.",
   "postalCode" : "48104-2201",
   "locality" : "Ann Arbor",
   "administrativeArea" : "MI",
   "firstName" : "Kyong-Jin",
   "email" : "test@cybs.com"
  },
  "amountDetails" : {
   "totalAmount" : "100.00",
   "currency" : "eur"
  }
 },
 "paymentInformation" : {
  "card" : {
   "expirationYear" : "2031",
   "number" : "4111111111111111",
   "expirationMonth" : "12"
  }
 }
}
```

Response to a Successful Request

```
{
  "_links": {
    "void": {
      "method": "POST",
      "href": "/pts/v2/credits/6663069906146706403954/voids"
    },
    "self": {
      "method": "GET",
      "href": "/pts/v2/credits/6663069906146706403954"
    }
  },
  "clientReferenceInformation": {
    "code": "1666306990717"
  },
  "creditAmountDetails": {
```

```
      "currency": "eur",
      "creditAmount": "100.00"
    },
    "id": "6663069906146706403954",
    "orderInformation": {
      "amountDetails": {
        "currency": "eur"
      }
    },
    "paymentAccountInformation": {
      "card": {
        "type": "001"
      }
    },
    "paymentInformation": {
      "tokenizedCard": {
        "type": "001"
      },
      "card": {
        "type": "001"
      }
    },
    "processorInformation": {
      "approvalCode": "888888",
      "responseCode": "100"
    },
    "reconciliationId": "66490108K9CLFJPN",
    "status": "PENDING",
    "submitTimeUtc": "2022-10-20T23:03:10Z"
}
```


Sales

This section provides the information you need in order to process a sale transaction. A sale transaction combines an authorization and a capture into a single transaction.

Endpoint

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

Required Fields for Processing a Sale

Use these required fields for processing a sale.

> **Important**
>
> When using relaxed requirements for address data and the expiration date, not all fields in this list are required. It is your responsibility to determine whether your account is enabled to use this feature and which fields are required. For details

about relaxed requirements, see *Relaxed Requirements for Address Data and Expiration Date in Payment Transactions* on page 264.

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.securityCode	
paymentInformation.card.type	
processingInformation.capture	**Set the value to** `true`.

Related Information

- *API field reference guide for the REST API*

REST Example: Processing a Sale

Endpoint:

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

Request

```
{
 "processingInformation": {
  "capture": true
 },
 "orderInformation" : {
  "billTo" : {
  "country" : "US",
  "lastName" : "VDP",
  "address1" : "201 S. Division St.",
  "postalCode" : "48104-2201",
  "locality" : "Ann Arbor",
```

```
    "administrativeArea" : "MI",
    "firstName" : "RTS",
    "email" : "test@cybs.com"
   },
    "amountDetails" : {
     "totalAmount" : "100.00",
     "currency" : "usd"
    }
   },
  "paymentInformation" : {
   "card" : {
    "expirationYear" : "2031",
    "number" : "4111111111111111",
    "expirationMonth" : "12",
    "type" : "001
   }
  }
}
```

Response to a Successful Request

Most processors do not return all of the fields that are shown in this example.

```
{
 "_links" : {
  "void" : {
   "method" : "POST",
   "href" : "/pts/v2/payments/6485004068966546103093/voids"
  },
  "self" : {
   "method" : "GET",
   "href" : "/pts/v2/payments/648500406896654610309 3"
  }
 },
 "clientReferenceInformation" : {
  "code" : "RTS-Auth"
 },
 "id" : "6485004068966546103093",
 "orderInformation" : {
  "amountDetails" : {
   "totalAmount" : "100.00",
   "authorizedAmount" : "100.00",
   "currency" : "usd"
  }
 },
 "paymentAccountInformation" : {
  "card" : {
   "type" : "001"
  }
 },
 "paymentInformation" : {
  "tokenizedCard" : {
   "type" : "001"
  },
  "card" : {
   "type" : "001"
```

```
    }
   },
   "processorInformation" : {
    "systemTraceAuditNumber" : "841109",
    "approvalCode" : "831000",
    "merchantAdvice" : {
     "code" : "01",
     "codeRaw" : "M001"
    },
    "responseDetails" : "ABC",
    "networkTransactionId" : "016153570198200",
    "retrievalReferenceNumber" : "208720841109",
    "consumerAuthenticationResponse" : {
     "code" : "2",
     "codeRaw" : "2"
    },
    "transactionId" : "016153570198200",
    "responseCode" : "00",
    "avs" : {
     "code" : "Y",
     "codeRaw" : "Y"
    }
   },
   "reconciliationId" : "64850040068966546103093",
   "status" : "AUTHORIZED",
   "submitTimeUtc" : "2022-03-28T20:46:47Z"
}
```


Void

This section describes how to void a capture or credit that was submitted but not yet processed by the processor.

Endpoint

Void a Capture

Production: `POST https://api.cybersource.com/pts/v2/captures/{id}/voids`

Test: `POST https://apitest.cybersource.com/pts/v2/captures/{id}/voids`

Void a Credit

Production: `POST https://api.cybersource.com/pts/v2/credits/{id}/voids`

Test: `POST https://apitest.cybersource.com/pts/v2/credits/{id}/voids`

The `{id}` is the transaction ID returned during the request to void.

Required Fields for Processing a Void

clientReferenceInformation.code

clientReferenceInformation.partner.thirdPartyCertificationNumber

Cybersource provides the value for this field.

id	**Set the id URL parameter to the request ID that was included in the authorization response message.**

REST Example: Processing a Void

Endpoint:

- Captures:
  - Production: POST https://api.cybersource.com/pts/v2/captures/{id}/voids
  - Test: POST https://apitest.cybersource.com/pts/v2/captures/{id}/voids
- Credits:
  - Production: POST https://api.cybersource.com/pts/v2/credits/{id}/voids
  - Test: POST https://apitest.cybersource.com/pts/v2/credits/{id}/voids

Important

A POST request for a void requires a body. If you have no fields to send, use empty braces as shown below. If you have fields to pass, see Example: Request with Fields, shown below.

Request with No Fields (Empty Braces)

```
{
}
```

Request with Fields

```
{
  "clientReferenceInformation": {
    "code": "test123",
    "partner": {
      "thirdPartyCertificationNumber": "testTPCN"
    }
  }
}
```

Response to a Successful Request

```
{
  "_links": {
    "self": {
      "method": "GET",
      "href": "/pts/v2/voids/6541933390746728203005"
    }
  },
  "clientReferenceInformation": {
     "code": "1654193339056"
  },
  "id": "6541933390746728203005",
  "orderInformation": {
```

```
    "amountDetails": {
      "currency": "USD"
    }
  },
  "status": "VOIDED",
  "submitTimeUtc": "2022-06-02T18:08:59Z",
  "voidAmountDetails": {
    "currency": "usd",
    "voidAmount": "100.00"
  }
}
```


Timeout Voids for a Capture, Sale, Refund, or Credit

When you do not receive a response message after sending a capture, sale, or credit request, this feature enables you to void the transaction that you requested.
Include the **clientReferenceInformation.transactionId** field in the original request for a capture, sale, refund, or credit. The value of the merchant transaction ID must be unique for 180 days.
When the original transaction fails, the response message for the reversal request includes these fields:

- **voidAmountDetails.originalTransactionAmount**
- **processorInformation.responseCode**

Endpoint

Production: POST https://api.cybersource.com/pts/v2/voids/
Test: POST https://apitest.cybersource.com/pts/v2/voids/

Required Fields for Processing a Timeout Void for a Capture, Sale, Refund, or Credit

Use the value from this field to request a timeout void:

clientReferenceInformation.transactionId	**Identifier that links the void request to the original request.**

Related Information

- *API field reference guide for the REST API*

REST Example: Processing a Timeout Void for a Capture, Sale, Refund, or Credit

Endpoint

- Production: POST https://api.cybersource.com/pts/v2/voids
- Test: POST https://apitest.cybersource.com/pts/v2/voids

Request

```
{
 "clientReferenceInformation": {
  "transactionId": "987654321"
 }
}
```

Response to a Successful Request

```
{
 "_links": {
  "self": {
   "method": "GET",
    "href": "/pts/v2/voids/6541933390746728203005"
  }
 },
 "clientReferenceInformation": {
  "code": "1654193339056"
 },
 "id": "6541933390746728203005",
 "orderInformation": {
  "amountDetails": {
   "currency": "USD"
  }
 },
 "status": "VOIDED",
 "submitTimeUtc": "2022-06-02T18:08:59Z",
 "voidAmountDetails": {
  "currency": "usd",
  "voidAmount": "100.00"
 }
}
```

Card Present Connect | Retail Processing

This section shows you how to process card-present transactions that use:

- Authorizations with contact or contactless EMV and an online PIN or an offline PIN
- Authorizations with magnetic stripe swipe
- Authorizations with hand-keyed data
- Authorizations for a cash advance
- Capture with a contact EMV

Additional Resources for Card Present Connect | Retail

For more information on Card Present Connect | Retail, see these guides:

- *Card Present Connect | Retail Guide*
- *API field reference guide for the REST API*
- Github repositories: *https://github.com/Cybersource?q=rest-sample&type=all&language=&sort=*

Authorization with Contact EMV and Online PIN

For an EMV chip contact authorization, the customer inserts the card directly into a point-of-sale (POS) terminal. For an online PIN authorization, the customer enters a PIN to verify their identity, and the issuer verifies the PIN.

Endpoint

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

Required Fields for Processing an Authorization with Contact EMV and Online PIN

clientReferenceInformation.code	
clientReferenceInformation.partner.thirdPartyCertificationNumber	**Cybersource provides the value for this field.**
clientReferenceInformation.transactionId	
merchantInformation.transactionLocalDateTime	
orderInformation.amountDetails.currency	
orderInformation.amountDetails.totalAmount	
paymentInformation.card.type	
pointOfSaleInformation.emv.cardSequenceNumber	
pointOfSaleInformation.emv.tags	
pointOfSaleInformation.encryptedKeySerialNumber	
pointOfSaleInformation.encryptedPin	
pointOfSaleInformation.entryMode	**Set the value to `contact` for an EMV payment.**
pointOfSaleInformation.pinBlockEncodingFormat	
pointOfSaleInformation.terminalCapability	**Set the value to `4`.**
pointOfSaleInformation.terminalPinCapability	
pointOfSaleInformation.trackData	
processingInformation.commerceIndicator	**Set the value to `retail`.**

Country-Specific Required Fields for Processing an Authorization with Contact EMV or Contactless PIN

Argentina

merchantInformation.transactionLocalDateTime	**Required when the time zone is not set in your account.**
invoiceDetails.salesSlipNumber	

India

pointOfSaleInformation.terminalCompliance

Japan

invoiceDetails.salesSlipNumber

REST Example: Processing an Authorization with Contact EMV and Online PIN

Endpoint:

- Production: POST https://api.cybersource.com/pts/v2/payments
- Test: POST https://apitest.cybersource.com/pts/v2/payments

Request


```
{
 "clientReferenceInformation": {
  "code": "test123",
  "transactionId": "uniqueValue1",
  "partner": {
   "thirdPartyCertificationNumber": "testTPCN"
  }
 },
 "processingInformation": {
  "commerceIndicator": "retail",
  }
 },
 "paymentInformation": {
   "card": {
         "type": "001"
    }
 },
 "orderInformation": {
  "amountDetails": {
   "totalAmount": "9900.00",
   "currency": "USD"
  }
 },
 "pointOfSaleInformation": {
  "entryMode": "contact",
  "terminalCapability": 4,
   "terminalPinCapability": 6,
  "emv": {
   "tags":
"5F3401019F3303E0F8C8950580800480009F370465B81A3A9F1007060112030A0A0009F2608E9D097D1901E8AB99F36020002
   "cardSequenceNumber": "01"
  },
  "trackData": ";4761739001010143=251220111478549?",
  "pinBlockEncodingFormat":0,
  "encryptedPin": "F509429A3C3FD201",
```

```
    "encryptedKeySerialNumber": "FFFF1B1D140000200001"
   },
    "merchantInformation": {
   "transactionLocalDateTime": "20230724085022"
  }
 }
```

Response to a Successful Request

```
{
  "_links": {
    "authReversal": {
      "method": "POST",
      "href": "/pts/v2/payments/6938891699856080004953/reversals"
    },
    "self": {
      "method": "GET",
      "href": "/pts/v2/payments/693889169985608004953"
    },
    "capture": {
      "method": "POST",
      "href": "/pts/v2/payments/6938891699856080004953/captures"
    }
  },
  "clientReferenceInformation": {
    "code": "test123",
    "transactionId": "uniqueValue1"
  },
  "id": "6938891699856080004953",
  "orderInformation": {
    "amountDetails": {
      "authorizedAmount": "9900.00",
      "currency": "USD"
    }
  },
  "paymentAccountInformation": {
    "card": {
      "type": "001"
    }
  },
  "paymentInformation": {
    "accountFeatures": {
      "category": "A",
      "group": "0"
    },
    "tokenizedCard": {
      "type": "001"
    },
    "card": {
      "type": "001"
    }
  },
  "pointOfSaleInformation": {
    "emv": {
      "tags":
"9F36020015910AB58D60185BEF0247303072179F1804303031860E04DA9F5809003B1BAEDFD1438BA48"
```

```
    }
   },
   "processorInformation": {
    "systemTraceAuditNumber": "188535",
    "approvalCode": "831000",
    "networkTransactionId": "016153570198200",
    "retrievalReferenceNumber": "324704188535",
    "transactionId": "016153570198200",
    "responseCode": "00",
    "avs": {
     "code": "2"
    }
   },
   "reconciliationId": "6938891699856080004953",
   "status": "AUTHORIZED",
   "submitTimeUtc": "2023-09-05T04:46:10Z"
}
```


Authorization with Contact EMV and Offline PIN

During a contact EMV authorization, the customer inserts the card into the terminal, which causes the EMV chip to be in contact with the terminal. When processing an offline PIN transaction, the EMV chip verifies the customer PIN.

Endpoint

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

Required Fields for Processing an Authorization with Contact EMV and Offline PIN

clientReferenceInformation.code	
clientReferenceInformation.partner.thirdPartyCertificationNumber	**Cybersource provides the value for this field.**
clientReferenceInformation.transactionId	
merchantInformation.transactionLocalDateTime	
orderInformation.amountDetails.currency	
orderInformation.amountDetails.totalAmount	
paymentInformation.card.type	
pointOfSaleInformation.emv.cardSequenceNumber	
pointOfSaleInformation.emv.tags	

pointOfSaleInformation.entryMode	**Set the value to `contact`.**
pointOfSaleInformation.terminalCapability	**Set the value to `4`. Set the value to `0` if the terminal does not support PINs.**
pointOfSaleInformation.terminalPinCapability	
pointOfSaleInformation.trackData	
processingInformation.commerceIndicator	**Set the value to `retail`.**

Country-Specific Required Fields for Processing an Authorization with Contact EMV or Contactless PIN

Argentina

merchantInformation.transactionLocalDateTime	**Required when the time zone is not set in your account.**
invoiceDetails.salesSlipNumber	

India

pointOfSaleInformation.terminalCompliance

Japan

invoiceDetails.salesSlipNumber

REST Example: Processing an Authorization with Contact EMV and Offline PIN

Endpoint:

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

Request

```
{
 "clientReferenceInformation": {
  "code": "test123",
  "transactionId": "uniqueValue2",
  "partner": {
   "thirdPartyCertificationNumber": "testTPCN"
  }
 },
 "processingInformation": {
  "commerceIndicator": "retail",
  "authorizationOptions": {
   "partialAuthIndicator": "true"
  }
```

```
  },
  "paymentInformation": {
    "card": {
          "type": "001"
     }
  },
  "orderInformation": {
   "amountDetails": {
    "totalAmount": "9900.00",
    "currency": "USD"
   }
  },
  "pointOfSaleInformation": {
   "entryMode": "contact",
   "terminalCapability": 4,
    "terminalPinCapability": 6,
   "emv": {
    "tags":
"5F3401019F3303E0F8C8950580800480009F370465B81A3A9F1007060112030A0A0009F2608E9D097D1901E8AB99F36020002
    "cardSequenceNumber": "01"
   },
   "trackData": ";4761739001010143=251220111478549?"
  },
   "merchantInformation": {
  "transactionLocalDateTime": "20230724085022"
 }
}
```

Response to a Successful Request

```
{
  "_links": {
   "authReversal": {
    "method": "POST",
    "href": "/pts/v2/payments/6938894575296498704951/reversals"
   },
   "self": {
    "method": "GET",
    "href": "/pts/v2/payments/693889457529649870 4951"
   },
   "capture": {
    "method": "POST",
    "href": "/pts/v2/payments/6938894575296498704951/captures"
   }
  },
  "clientReferenceInformation": {
   "code": "test123",
   "transactionId": "uniqueValue2"
  },
  "id": "6938894575296498704951",
  "orderInformation": {
   "amountDetails": {
    "authorizedAmount": "9900.00",
    "currency": "USD"
   }
  },
```

```
  "paymentAccountInformation": {
   "card": {
    "type": "001"
   }
  },
  "paymentInformation": {
   "accountFeatures": {
    "category": "A",
    "group": "0"
   },
   "tokenizedCard": {
    "type": "001"
   },
   "card": {
    "type": "001"
   }
  },
  "pointOfSaleInformation": {
   "emv": {
    "tags":
 "9F36020015910AB58D60185BEF0247303072179F18043030303186 0E04DA9F5809 03B1BAEDFD1438BA48"
   }
  },
  "processorInformation": {
   "systemTraceAuditNumber": "188589",
   "approvalCode": "831000",
   "networkTransactionId": "016153570198200",
   "retrievalReferenceNumber": "324704188589",
   "transactionId": "016153570198200",
   "responseCode": "00",
   "avs": {
    "code": "2"
   }
  },
  "reconciliationId": "6938894575296498704951",
  "status": "AUTHORIZED",
  "submitTimeUtc": "2023-09-05T04:50:58Z"
}
```


Authorization with Contactless EMV and Online PIN

For an EMV contactless payment, the customer taps the card on the terminal. The terminal and chip use near-field communication (NFC) to communicate with each other. For an online PIN transaction, the customer uses a PIN to verify their identity and the issuer verifies the PIN.

Endpoint

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

Required Fields for Processing an Authorization with Contactless EMV and Online PIN

clientReferenceInformation.code	
clientReferenceInformation.partner.thirdPartyCertificationNumber	Cybersource provides the value for this field.
clientReferenceInformation.transactionId	
merchantInformation.transactionLocalDateTime	
orderInformation.amountDetails.currency	
orderInformation.amountDetails.totalAmount	
paymentInformation.card.type	
pointOfSaleInformation.emv.cardSequenceNumber	
pointOfSaleInformation.emv.tags	
pointOfSaleInformation.encryptedKeySerialNumber	
pointOfSaleInformation.encryptedPin	
pointOfSaleInformation.entryMode	Set the value to `contactless` for an EMV payment.
pointOfSaleInformation.pinBlockEncodingFormat	
pointOfSaleInformation.terminalCapability	Set the value to `5`.
pointOfSaleInformation.terminalPinCapability	
pointOfSaleInformation.trackData	
processingInformation.commerceIndicator	Set the value to `retail`.

REST Example: Processing an Authorization with Contactless EMV and Online PIN

Endpoint:

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

Request

```
{
 "clientReferenceInformation": {
  "code": "test123",
  "transactionId": "uniqueValue3",
  "partner": {
   "thirdPartyCertificationNumber": "testTPCN"
  }
 },
 "processingInformation": {
```

```
   "commerceIndicator": "retail",
   "authorizationOptions": {
    "partialAuthIndicator": "true"
   }
  },
  "paymentInformation": {
    "card": {
      "type": "001"
     }
  },
  "orderInformation": {
   "amountDetails": {
    "totalAmount": "9900.00",
    "currency": "USD"
   }
  },
  "pointOfSaleInformation": {
   "entryMode": "contactless",
   "terminalCapability": 4,
    "terminalPinCapability": 6,
   "emv": {
    "tags":
 "5F3401019F3303E0F8C8950580800480009F370465B81A3A9F1007060112030A0A0009F2608E9D097D1901E8AB99F360200025
    "cardSequenceNumber": "01"
   },
   "trackData": ";4761739001010143=251220111478549?",
   "pinBlockEncodingFormat":0,
   "encryptedPin": "F509429A3C3FD201",
   "encryptedKeySerialNumber": "FFFF1B1D140000200001"
  },
   "merchantInformation": {
  "transactionLocalDateTime": "20230724085022"
 }
}
```

Response to a Successful Request

```
{
  "_links": {
   "authReversal": {
    "method": "POST",
    "href": "/pts/v2/payments/6938904668436727104951/reversals"
   },
   "self": {
    "method": "GET",
    "href": "/pts/v2/payments/6938904668436727104951"
   },
   "capture": {
    "method": "POST",
    "href": "/pts/v2/payments/6938904668436727104951/captures"
   }
  },
  "clientReferenceInformation": {
   "code": "test123",
   "transactionId": "uniqueValue3"
  },
```

```
  "id": "6938904668436727104951",
  "orderInformation": {
    "amountDetails": {
      "authorizedAmount": "9900.00",
      "currency": "USD"
    }
  },
  "paymentAccountInformation": {
    "card": {
      "type": "001"
    }
  },
  "paymentInformation": {
    "accountFeatures": {
      "category": "A",
      "group": "0"
    },
    "tokenizedCard": {
      "type": "001"
    },
    "card": {
      "type": "001"
    }
  },
  "pointOfSaleInformation": {
    "emv": {
      "tags":
"9F36020015910AB58D60185BEF0247303072179F1804303031860E04DA9F5809031B1BAEDFD1438BA48"
    }
  },
  "processorInformation": {
    "systemTraceAuditNumber": "188851",
    "approvalCode": "831000",
    "networkTransactionId": "016153570198200",
    "retrievalReferenceNumber": "324705188851",
    "transactionId": "016153570198200",
    "responseCode": "00",
    "avs": {
      "code": "2"
    }
  },
  "reconciliationId": "6938904668436727104951",
  "status": "AUTHORIZED",
  "submitTimeUtc": "2023-09-05T05:07:47Z"
}
```


Authorizations with Magnetic Stripe Swipes

Although EMV chips have become common, sometimes the EMV chip cannot be used to validate the customer. In these instances, you can choose to validate the customer by using the magnetic stripe on the payment card.

Endpoint

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

Required Fields for Processing an Authorization with Swiped Track Data

clientReferenceInformation.code	
clientReferenceInformation.partner.thirdPartyCertificationNumber	**Cybersource provides the value for this field.**
clientReferenceInformation.transactionId	
merchantInformation.transactionLocalDateTime	
orderInformation.amountDetails.currency	
orderInformation.amountDetails.totalAmount	
paymentInformation.card.type	
pointOfSaleInformation.entryMode	**Set the value to `swiped`.**
pointOfSaleInformation.terminalCapability	
pointOfSaleInformation.terminalPinCapability	
pointOfSaleInformation.trackData	
processingInformation.commerceIndicator	**Set the value to `retail`**

REST Example: Processing an Authorization with Swiped Track Data

Endpoint:

- Production: POST https://api.cybersource.com/pts/v2/payments
- Test: POST https://apitest.cybersource.com/pts/v2/payments

Request

```
{
  "clientReferenceInformation": {
    "code": "ABC123",
    "partner": {
      "thirdPartyCertificationNumber": "123456789012"
    }
  },
  "processingInformation": {
    "commerceIndicator": "retail"
    }
  },
  "pointOfSaleInformation": {
    "trackData": ";4111111111111111=2312201123456678?",
    "entryMode": "swiped",
    "terminalCapability": "4"
```

```
  },
  "paymentInformation": {
    "card": {
      "type": "001"
    }
  },
  "orderInformation": {
    "amountDetails": {
      "totalAmount": "9601.00",
      "currency": "USD"
    }
  },
}
```

Response to a Successful Request

```
{
  "_links": {
    "authReversal": {
      "method": "POST",
      "href": "/pts/v2/payments/6869553167546562203955/reversals"
    },
    "self": {
      "method": "GET",
      "href": "/pts/v2/payments/686955316754656220 3955"
    },
    "capture": {
      "method": "POST",
      "href": "/pts/v2/payments/6869553167546562203955/captures"
    }
  },
  "clientReferenceInformation": {
    "code": "ABC123"
  },
  "id": "6869553167546562203955",
  "orderInformation": {
    "amountDetails": {
      "authorizedAmount": "9601.00",
      "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": "1"
    }
  },
  "reconciliationId": "63427009RIT9HBR9",
  "status": "AUTHORIZED",
  "submitTimeUtc": "2023-06-16T22:41:57Z"
}
```


Authorizations with Hand-Keyed Data

Under certain circumstances, you might choose to manually enter (hand key) a customer's data to obtain an authorization.

Endpoint

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

Required Fields for Processing an Authorization with Hand Keyed Data

Important

When using relaxed requirements for address data and the expiration date, not all fields in this list are required. It is your responsibility to determine whether your account is enabled to use this feature and which fields are required. For details about relaxed requirements, see *Relaxed Requirements for Address Data and Expiration Date in Payment Transactions* on page 264.

clientReferenceInformation.code

clientReferenceInformation.partner.thirdPartyCertificationNumber
Cybersource provides the value for this field.

clientReferenceInformation.transactionId

merchantInformation.transactionLocalDateTime

orderInformation.amountDetails.currency

orderInformation.amountDetails.totalAmount

paymentInformation.card.expirationMonth

paymentInformation.card.expirationyear	
paymentInformation.card.number	
pointOfSaleInformation.cardPresent	**Set the value to `true`.**
pointOfSaleInformation.entryMode	**Set the value to `keyed`.**
pointOfSaleInformation.terminalCapability	**Must be set to 1, 2, 3, 4, or 5.**
pointOfSaleInformation.terminalPinCapability	
processingInformation.commerceIndicator	**Set the value to `retail`.**

REST Example: Processing an Authorization with Hand Keyed Data

Endpoint:

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

Request

```
{
  "clientReferenceInformation": {
    "code": "123456",
    "transactionId": "12233445679",
    "partner": {
      "thirdPartyCertificationNumber": "123456789012"
    }
  },
  "processingInformation": {
    "commerceIndicator": "retail",
    "authorizationOptions": {
      "ignoreAvsResult": "true",
      "ignoreCvResult": "true"
    }
  },
  "pointOfSaleInformation": {
    "entryMode": "keyed",
    "terminalCapability": "4",
    "terminalPinCapability": "6"
  },
  "paymentInformation": {
    "card": {
      "number": "4111111111111111",
      "securityCode": "123",
      "expirationMonth": "12",
      "expirationYear": "2031",
      "type": "001"
    }
  },
  "orderInformation": {
    "amountDetails": {
      "totalAmount": "9604.00",
      "currency": "USD"
    },
```

```
    "billTo": {
      "postalCode": "94538"
    }
  }
  "merchantInformation": {
    "transactionLocalDateTime": "20230724085022"
  }
}
```

Response to a Successful Request

A successful response returns `status=AUTHORIZED`.

```
{
  "_links": {
    "void": {
      "method": "POST",
      "href": "/pts/v2/payments/6080032225246314603005/voids"
    },
    "self": {
      "method": "GET",
      "href": "/pts/v2/payments/6080032225246314603005"
    }
  },
  "clientReferenceInformation": {
    "code": "123456"
  },
  "id": "6080032225246314603005",
  "orderInformation": {
    "amountDetails": {
      "totalAmount": "9604.00",
      "authorizedAmount": "9604.00",
      "currency": "USD"
    }
  },
  "paymentAccountInformation": {
    "card": {
      "type": "001"
    }
  },
  "paymentInformation": {
    "accountFeatures": {
      "category": "A",
      "group": "0"
    },
    "tokenizedCard": {
      "type": "001"
    }
  },
  "processorInformation": {
    "systemTraceAuditNumber": "173156",
    "approvalCode": "831000",
    "cardVerification": {
      "resultCodeRaw": "M",
      "resultCode": "M"
    },
```

```
    "networkTransactionId": "016153570198200",
    "transactionId": "016153570198200",
    "responseCode": "00",
    "avs": {
        "code": "Z",
        "codeRaw": "Z"
    }
},
"reconciliationId": "60800322252463146030005",
"status": "AUTHORIZED",
"submitTimeUtc": "2020-12-15T03:33:42Z"
}
```


Authorization for a Cash Advance with a Credit Card

With a cash advance, a cardholder can withdraw cash against their credit card account limit at an ATM or at their bank. At the ATM, the cardholder is required to enter a PIN, which they may be allowed to request online. At the bank, the cardholder hands their credit card and identification to the bank clerk, or uses the bank card terminal.
This feature is supported for Discover, Mastercard, and Visa credit cards in the U.S.

Field Specific to This Use Case

This API field is specific to this use case:

processingInformation.authorizationOptions.cashAdvance **Set the value to** `true`.

Endpoint

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

Required Fields for Processing an Authorization for a Cash Advance

Use these required fields for processing an authorization for a cash advance.

Important

When using relaxed requirements for address data and the expiration date, not all fields in this list are required. It is your responsibility to determine whether your account is enabled to use this feature and which fields are required. For details about relaxed requirements, see *Relaxed Requirements for Address Data and Expiration Date in Payment Transactions* on page 264.

merchantInformation.categoryCode	**Set the value to `6010` for cash advance authorization.**

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

processingInformation.authorizationOptions.isCashAdvanceIndicator **Set the value to** `true`.

Related Information

- *API field reference guide for the REST API*

Optional Fields for Processing an Authorization for a Cash Advance

You can use these optional fields to include additional information when processing an authorization for a cash advance.

clientReferenceInformation.code

orderInformation.billTo.address2

orderInformation.billTo.buildingNumber

orderInformation.billTo.company.name

orderInformation.billTo.district

paymentInformation.card.securityCode

Related Information

- *API field reference guide for the REST API*

REST Example: Processing an Authorization for a Cash Advance

Endpoint:

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

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

This example shows a cash advance request against a Mastercard debit card account limit.

Request

```
{
  "clientReferenceInformation": {
    "code": "TC50171rbi_2"
  },
  "processorInformation": {
    "authorizationOptions": {
      "panReturnIndicator": "1-E"
    }
  },
  "orderInformation": {
    "billTo": {
      "country": "US",
      "lastName": "VDP",
      "address2": "test",
      "address1": "201 S. Division St.",
      "postalCode": "48104-2201",
      "locality": "Ann Arbor",
      "administrativeArea": "MI",
      "firstName": "RTS",
      "phoneNumber": "999999999",
      "district": "MI",
      "buildingNumber": "123",
      "company": "Visa",
      "email": "test@cybs.com"
    },
    "amountDetails": {
      "totalAmount": "6854.31",
      "currency": "USD"
    }
  },
  "paymentInformation": {
    "card": {
      "expirationYear": "2031",
      "number": "5555555555554444",
      "securityCode": "123",
      "expirationMonth": "12",
      "type": "002",
      "useAs": "DB"
    }
  },
  "buyerInformation": {
    "personalIdentification": [
      {
        "type": "TAX_ID",
        "issuedBy": "issuer",
        "verificationResults": "5"
      }
    ]
  },
  "processingInformation": {
    "authorizationOptions": {
```

```
        "cashAdvanceIndicator": "true"
      }
    }
}
```

Response to a Successful Request

```
{
  "_links": {
    "authReversal": {
      "method": "POST",
      "href": "/pts/v2/payments/6557018740636000603187/reversals"
    },
    "self": {
      "method": "GET",
      "href": "/pts/v2/payments/655701874063600060 3187"
    },
    "capture": {
      "method": "POST",
      "href": "/pts/v2/payments/6557018740636000603187/captures"
    }
  },
  "clientReferenceInformation": {
    "code": "TC50171rbi_2"
  },
  "id": "6557018740636000603187",
  "orderInformation": {
    "amountDetails": {
      "authorizedAmount": "6854.31",
      "currency": "USD"
    }
  },
  "paymentAccountInformation": {
    "card": {
      "type": "002"
    }
  },
  "paymentInformation": {
    "tokenizedCard": {
      "type": "002"
    },
    "card": {
      "hashedNumber": "555555L3JbvR9AWh7QM2q6+F3f62kCqZhKdv2HfDtcw7UIWoI=",
      "type": "002"
    }
  },
  "processorInformation": {
    "systemTraceAuditNumber": "814983",
    "electronicVerificationResults": {
      "codeRaw": "01"
    },
    "approvalCode": "831000",
    "cardVerification": {
      "resultCodeRaw": "M",
      "resultCode": "M"
    },
```

```
      "merchantAdvice": {
        "code": "01",
        "codeRaw": "M001",
        "nameMatch": "00"
      },
      "networkTransactionId": "MCC6635490620",
      "retrievalReferenceNumber": "217105814983",
      "consumerAuthenticationResponse": {
        "code": "2",
        "codeRaw": "2"
      },
      "transactionId": "MCC6635490620",
      "responseCode": "00",
      "avs": {
        "code": "Y",
        "codeRaw": "Y"
      }
    },
    "reconciliationId": "65570187406360006O3187",
    "status": "AUTHORIZED",
    "submitTimeUtc": "2022-06-20T05:11:14Z"
}
```


Response Codes from an Authorization for a Cash Advance

201	**Successful response.**
400	**Invalid request.**
502	**Unexpected system error or system timeout.**

Captures

This section provides the information you need in order to capture an authorized transaction.

Endpoint

Production: `POST https://api.cybersource.com/pts/v2/payments/{id}/captures`
Test: `POST https://apitest.cybersource.com/pts/v2/payments/{id}/captures`
The `{id}` is the transaction ID returned in the authorization response.

Required Fields for Capturing an Authorization

Use these required fields for capturing an authorization.

clientReferenceInformation.code	**This field value maps from the original authorization, sale, or credit transaction.**

clientReferenceInformation.partner.thirdPartyCertificationNumber	**Cybersource provides the value for this field.**
orderInformation.amountDetails.currency	
orderInformation.amountDetails.totalAmount	

Capturing an Authorization Using REST APIs

1. Pass the original authorization ID in the URL, and send the service request to:

   `POST https://<url_prefix>/v2/payments/{id}/captures`
   Use one of these URL prefixes:

   - Test: `apitest.cybersource.com`
   - Production: `api.cybersource.com`
   - Production in India: `api.in.cybersource.com`

   Where `id` is the authorization ID returned in the authorization response.

   ```
   {
    "id": "6481692924466004003001"
   }
   ```

   The URL with the `id` value is included in the authorization response:

   ```
   {
    "_links": {
     "capture": {
      "method": "POST",
      "href": "/pts/v2/payments/6481692924466004003001/captures"
     }
    }
   ```

2. Check the response message to make sure that the request was successful. A 200-level HTTP response code indicates success. For information about response codes, see *Transaction Response Codes*.

REST Example: Capturing an Authorization

Endpoint:

- Production: `POST https://api.cybersource.com/pts/v2/payments/{id}/captures`
- Test: `POST https://apitest.cybersource.com/pts/v2/payments/{id}/captures`

Request

```
{
  "clientReferenceInformation": {
    "code": "ABC123",
    "partner": {
      "thirdPartyCertificationNumber": "123456789012"
    }
  },
```

```
  "orderInformation": {
    "amountDetails": {
      "totalAmount": "100.00",
      "currency": "EUR"
  }
}
```

Response to a Successful Request

```
{
  "_links": {
    "void": {
      "method": "POST",
      "href": "/pts/v2/captures/6662994431376681303954/voids"
    },
    "self": {
      "method": "GET",
      "href": "/pts/v2/captures/666299443137668130 3954"
    }
  },
  "clientReferenceInformation": {
    "code": "1666299443215"
  },
  "id": "6662994431376681303954",
  "orderInformation": {
    "amountDetails": {
      "totalAmount": "100.00",
      "currency": "EUR"
    }
  },
  "reconciliationId": "66535942B9CGT52U",
  "status": "PENDING",
  "submitTimeUtc": "2022-10-20T20:57:23Z"
}
```

Card Present Connect | Mass Transit Processing

This section shows you how to process card-present transactions for Mass Transit.

Additional Resources for Card Present Connect | Mass Transit

For more information on Card Present Connect | Mass Transit:

- *Card Present Connect | Mass Transit Developer Guide*
- *API Field Reference Using the REST API*
- Github repositories: *https://github.com/Cybersource?q=rest-sample&type=all&language=&sort=*

Mastercard Authorization with EMV Data

A Mastercard authorization with EMV data is an authorization request for a nominal amount.

Endpoint

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

Example: Mastercard Authorization with EMV Data

Request

```
{
  "clientReferenceInformation": {
    "comments": "TransitDA BAU nominal value auth",
```

```
      "code": "10000568",
      "transactionId": "20000568",
      "partner": {
        "thirdPartyCertificationNumber": "BPCDRC220403",
        "solutionId": "548UHQ8Z"
      }
    },
    "processingInformation": {
      "industryDataType": "transit",
      "commerceIndicator": "retail",
      "capture": "false",
      "captureOptions": {
        "dateToCapture": "0425"
      },
      "authorizationOptions": {
        "authIndicator": "0",
        "deferredAuthIndicator": "true",
        "aggregatedAuthIndicator": "true",
        "transportationMode": "00"
      }
    },
    "orderInformation": {
      "amountDetails": {
        "totalAmount": "10.00",
        "currency": "EUR"
      }
    },
    "paymentInformation": {
      "card": {
        "type": "002"
      },
      "initiationChannel": "00"
    },
    "pointOfSaleInformation": {
      "terminalId": "12345678",
      "catLevel": "2",
      "entryMode": "contactless",
      "terminalCapabiity": "5",
        "terminalPinCapability": "0",
      "emv": {
        "tags":
 "5F2A0209768407A0000000004101 09F3602000 39F03060000000000009C01005F3401019F10120110A0000F040000000000
      },
      "trackData": ";5413330089700042=49122010123456789?",
      "serviceCode": "201"
    }
}
```

Response to a Successful Request

```
{
    "_links": {
      "self": {
        "method": "GET",
        "href": "/pts/v2/payments/6508877845426512004004"
      }
```

```
    },
    "clientReferenceInformation": {
      "code": "10000574",
      "partner": {
        "solutionId": "548UHQ8Z"
      },
      "transactionId": "20000574"
    },
    "errorInformation": {
      "reason": "AUTH_DECLINE_CAPTURE_POSSIBLE",
      "message": "Authorization Declined. Follow-on Capture can be processed."
    },
    "id": "6508877845426512004004",
    "pointOfSaleInformation": {
      "emv": {
        "tags":
 "9F36020015910AB58D60185BEF0247303072179F1804303031860E04DA9F5803B1BAEDFD1438BA48"
      }
    },
    "processorInformation": {
      "systemTraceAuditNumber": "164207",
      "networktransactionId": "016153570198200",
      "retrievalReferenceNumber": "211511164207",
      "transactionId": "016153570198200",
      "responseCode": "05",
      "avs": {
        "code": "2"
      }
    },
    "status": "AUTHORIZED"
}
```


Visa Account Verification Request (AVR) with EMV Data

A Visa account verification request (AVR) with EMV data is a zero amount authorization.

Endpoint

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

Example: Visa AVR Authorization with EMV Data

Request


```
{
  "clientReferenceInformation": {
    "comments": "TransitDA BAU zero value auth",
    "code": "10000564",
    "transactionId": "20000564",
```

```
      "partner": {
        "thirdPartyCertificationNumber": "BPCDRC220403",
        "solutionId": "548UHQ8Z"
      }
    },
    "processingInformation": {
      "capture": "false",
      "commerceIndicator": "retail"
    },
    "orderInformation": {
      "amountDetails": {
        "totalAmount": "0.00",
        "currency": "EUR"
      }
    },
    "paymentInformation": {
      "card": {
        "type": "001"
      }
    },
    "pointOfSaleInformation": {
      "terminalId": "12345678",
      "catLevel": "2",
      "entryMode": "contactless",
      "terminalCapabiity": "5",
        "terminalPinCapability": "0",
      "emv": {
        "tags":
 "5F2A0209768407A0000000003101 09F3602000 29F0306000000000000 9C01005F3401019F10201F220100A000000000000000
        "cardSequenceNumber": "1"
      },
      "trackData": ";4761731000000027=241220119058254?"
    }
}
```


Response to a Successful Request


```
{
    "_links": {
      "authReversal": {
        "method": "POST",
        "href": "/pts/v2/payments/650887546612653810400 2/reversals"
      },
      "self": {
        "method": "GET",
        "href": "/pts/v2/payments/650887546612653810400 2"
      },
      "capture": {
        "method": "POST",
        "href": "/pts/v2/payments/650887546612653810400 2/captures"
      }
    },
    "clientReferenceInformation": {
      "code": "10000564",
      "partner": {
        "solutionId": "548UHQ8Z"
```

```
      },
      "transactionId": "20000564"
   },
   "id": "6508875466126538104002",
   "orderInformation": {
      "amountDetails": {
         "authorizedAmount": "0.00",
         "currency": "EUR"
      }
   },
   "paymentAccountInformation": {
      "card": {
         "type": "001"
      }
   },
   "paymentInformation": {
      "tokenizedCard": {
         "type": "001"
      },
      "card": {
         "type": "001"
      }
   },
   "processorInformation": {
      "systemTraceAuditNumber": "162930",
      "approvalCode": "831000",
      "merchantAdvice": {
         "code": "01",
         "codeRaw": "M001"
      },
      "responseDetails": "ABC",
      "networktransactionId": "016153570198200",
      "retrievalReferenceNumber": "211511162930",
      "consumerAuthenticationResponse": {
         "code": "2",
         "codeRaw": "2"
      },
      "transactionId": "016153570198200",
      "responseCode": "00",
      "avs": {
         "code": "Y",
         "codeRaw": "Y"
      }
   },
   "reconciliationId": "6508875466126538104002",
   "status": "AUTHORIZED",
   "submitTimeUtc": "2022-04-25T11:52:26Z"
}
```


Response to a Declined Request


```
{
   "_links": {
      "self": {
         "method": "GET",
         "href": "/pts/v2/payments/6508876049646556304003"
```

```
      }
   },
   "clientReferenceInformation": {
      "code": "10000566",
      "partner": {
         "solutionId": "548UHQ8Z"
      },
      "transactionId": "20000566"
   },
   "errorInformation": {
      "reason": "AUTH_DECLINE_CAPTURE_POSSIBLE",
      "message": "Authorization Declined. Follow-on Capture can be processed."
   },
   "id": "6508876049646556304003",
   "pointOfSaleInformation": {
      "emv": {
         "tags":
 "9F36020015910AB58D60185BEF0247303072179F18043030303186 0E04DA9F5803B1BAEDFD1438BA48"
      }
   },
   "processorInformation": {
      "systemTraceAuditNumber": "162936",
      "networktransactionId": "016153570198200",
      "retrievalReferenceNumber": "211511162936",
      "transactionId": "016153570198200",
      "responseCode": "05",
      "avs": {
         "code": "2"
      }
   },
   "status": "AUTHORIZED"
}
```

Visa Deferred Sale with EMV Data

A sale transaction is a bundled authorization and capture. At the end of the travel period, request a Visa deferred sale with EMV data for an aggregated payment.

Endpoint

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

Example: Visa Deferred Sale with EMV Data

Request

```
{
   "clientReferenceInformation": {
      "comments": "TransitDA BAU full value sale",
      "code": "10000565",
      "transactionId": "20000565",
```

```
      "partner": {
        "thirdPartyCertificationNumber": "BPCDRC220403",
        "solutionId": "548UHQ8Z"
      }
    },
    "processingInformation": {
      "industryDataType": "transit",
      "processingInformation.commerceIndicator": "retail",
      "capture": "true",
      "captureOptions": {
        "dateToCapture": "0425"
      },
      "authorizationOptions": {
        "deferredAuthIndicator": "true",
        "aggregatedAuthIndicator": "true"
      }
    },
    "orderInformation": {
      "amountDetails": {
        "totalAmount": "10.00",
        "currency": "EUR"
      }
    },
    "paymentInformation": {
      "card": {
        "type": "001"
      }
    },
    "pointOfSaleInformation": {
      "terminalId": "12345678",
      "catLevel": "2",
      "entryMode": "contactless",
      "terminalCapabiity": "5",
        "terminalPinCapability": "0",
      "emv": {
        "tags":
 "5F2A0209768407A0000000003101 09F360200029F03060000000000009C01005F3401019F10201F220100A000000000000
        "cardSequenceNumber": "1"
      },
      "trackData": ";4761731000000027=241220119058254?"
    }
}
```

Response to a Successful Request

```
{
    "_links": {
      "void": {
        "method": "POST",
        "href": "/pts/v2/payments/6508875814676551204001/voids"
      },
      "self": {
        "method": "GET",
        "href": "/pts/v2/payments/650887581467655120401"
      }
    },
```

```
  "clientReferenceInformation": {
    "code": "10000565",
    "partner": {
      "solutionId": "548UHQ8Z"
    },
    "transactionId": "20000565"
  },
  "id": "6508875814676551204001",
  "orderInformation": {
    "amountDetails": {
      "totalAmount": "10.00",
      "authorizedAmount": "10.00",
      "currency": "EUR"
    }
  },
  "paymentAccountInformation": {
    "card": {
      "type": "001"
    }
  },
  "paymentInformation": {
    "accountFeatures": {
      "group": "0"
    },
    "tokenizedCard": {
      "type": "001"
    },
    "card": {
      "type": "001"
    }
  },
  "pointOfSaleInformation": {
    "emv": {
      "tags":
"9F36020015910AB58D60185BEF0247303072179F18043030303186 0E04DA9F5809 03B1BAEDFD1438BA48"
    }
  },
  "processorInformation": {
    "systemTraceAuditNumber": "164186",
    "approvalCode": "831000",
    "networktransactionId": "016153570198200",
    "retrievalReferenceNumber": "211511164186",
    "transactionId": "016153570198200",
    "responseCode": "00",
    "avs": {
      "code": "2"
    }
  },
  "reconciliationId": "650887581467655120400 1",
  "status": "AUTHORIZED",
  "submitTimeUtc": "2022-04-25T11:53:01Z"
}
```


Response to a Declined Request with First Ride Protection


```
{
```

```
  "_links": {
    "self": {
      "method": "GET",
      "href": "/pts/v2/payments/6508878333386555704002"
    }
  },
  "clientReferenceInformation": {
    "code": "10000576",
    "partner": {
      "solutionId": "548UHQ8Z"
    },
    "transactionId": "20000576"
  },
  "errorInformation": {
    "reason": "AUTH_DECLINE_CAPTURE_POSSIBLE",
    "message": "Authorization Declined. Follow-on Capture can be processed."
  },
  "id": "6508878333386555704002",
  "pointOfSaleInformation": {
    "emv": {
      "tags":
"9F36020015910AB58D60185BEF0247303072179F1804303031860E04DA9F5809B1BAEDFD1438BA48"
    }
  },
  "processorInformation": {
    "systemTraceAuditNumber": "164212",
    "networktransactionId": "016153570198200",
    "retrievalReferenceNumber": "211511164212",
    "transactionId": "016153570198200",
    "responseCode": "05",
    "avs": {
      "code": "2"
    }
  },
  "status": "DECLINED"
}
```


Tap-Initiated Authorization for Debt Recovery with EMV Data

When a cardholder attempts to use a blocked card at the transit reader, create a new debt recovery authorization request using the chip data from the new tap, along with the fare amount of the previous declined authorization.

Endpoint

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

Example: Tap-Initiated Authorization for Debt Recovery with EMV Data

Request


```
{
  "clientReferenceInformation": {
    "comments": "TransitDA Debt recovery tap auth",
    "code": "10000597",
    "transactionId": "20000597",
    "partner": {
      "thirdPartyCertificationNumber": "BPCDRC220403",
      "solutionId": "548UHQ8Z"
    }
  },
  "processingInformation": {
    "industryDataType": "transit",
    "captureOptions": {
      "dateToCapture": "0425"
    },
    "capture": "false",
    "commerceIndicator": "retail",
    "authorizationOptions": {
      "debtRecoveryIndicator": "true",
      "deferredAuthIndicator": "true"
    }
  },
  "orderInformation": {
    "amountDetails": {
      "totalAmount": "10.00",
      "currency": "EUR"
    }
  },
  "paymentInformation": {
    "card": {
      "type": "001"
    }
  },
  "pointOfSaleInformation": {
    "terminalId": "12345678",
    "catLevel": "2",
    "entryMode": "contactless",
    "terminalCapabiity": "5",
      "terminalPinCapability": "0",
    "emv": {
      "tags":
 "5F2A0209768407A0000000003101 09F360200029F0306000000000000009C01005F3401019F10201F220100A000000000000
      "cardSequenceNumber": "1"
    },
    "trackData": ";4761731000000027=241220119058254?"
  }
}
```

Response to a Declined Request


```
{
  "_links": {
    "self": {
      "method": "GET",
      "href": "/pts/v2/payments/6508883585936636904003"
    }
  },
  "clientReferenceInformation": {
    "code": "10000597",
    "partner": {
      "solutionId": "548UHQ8Z"
    },
    "transactionId": "20000597"
  },
  "errorInformation": {
    "reason": "PROCESSOR_DECLINED",
    "message": "Decline - General decline of the card. No other information provided by the issuing bank."
  },
  "id": "6508883585936636904003",
  "pointOfSaleInformation": {
    "emv": {
      "tags":
"9F36020015910AB58D60185BEF0247303072179F18043030303186 0E04DA9F5803B1BAEDFD1438BA48"
    }
  },
  "processorInformation": {
    "systemTraceAuditNumber": "163648",
    "networktransactionId": "016153570198200",
    "retrievalReferenceNumber": "211512163648",
    "transactionId": "016153570198200",
    "responseCode": "05",
    "avs": {
      "code": "2"
    }
  },
  "status": "DECLINED"
}
```


Merchant-Initiated Authorizations for Debt Recovery with Stored Card Data

You can request the authorization service for a merchant-initiated debt recovery with stored card data.

Endpoint

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

Example: Merchant-Initiated Authorization for Debt Recovery with Stored Card Data

Request

```
{
  "clientReferenceInformation": {
    "comments": "TransitDA Debt recovery MIT auth",
    "code": "10000596",
    "transactionId": "20000596",
    "partner": {
      "thirdPartyCertificationNumber": "BPCDRC220403",
      "solutionId": "548UHQ8Z"
    }
  },
  "processingInformation": {
    "commerceIndicator": "moto",
    "industryDataType": "transit",
    "capture": "false",
    "authorizationOptions": {
      "debtRecoveryIndicator": "true",
      "ignoreAvsResult": "true",
      "ignoreCvResult": "true",
      "initiator": {
        "type": "merchant",
        "credentialStoredOnFile": "false",
        "storedCredentialUsed": "true",
        "merchantInitiatedTransaction": {
          "reason": "1",
          "previoustransactionId": "016153570198200"
        }
      }
    }
  },
  "paymentInformation": {
    "card": {
      "number": "4761731000000027",
      "expirationMonth": "12",
      "expirationYear": "2024",
      "type": "001"
    }
  },
  "orderInformation": {
    "amountDetails": {
      "totalAmount": "10.00",
      "currency": "EUR"
    }
  }
}
```

Response to a Declined Request

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

```
        "method": "GET",
        "href": "/pts/v2/payments/6508883374816631904001"
      }
    },
    "clientReferenceInformation": {
      "code": "10000596",
      "partner": {
        "solutionId": "548UHQ8Z"
      },
      "transactionId": "20000596"
    },
    "errorInformation": {
      "reason": "PROCESSOR_DECLINED",
      "message": "Decline - General decline of the card. No other information provided by the issuing bank."
    },
    "id": "6508883374816631904001",
    "pointOfSaleInformation": {
      "emv": {
        "tags":
 "9F36020015910AB58D60185BEF0247303072179F18043030303186 0E04DA9F5809 03B1BAEDFD1438BA48"
      }
    },
    "processorInformation": {
      "systemTraceAuditNumber": "164869",
      "networktransactionId": "016153570198200",
      "retrievalReferenceNumber": "211512164869",
      "transactionId": "016153570198200",
      "responseCode": "05",
      "avs": {
        "code": "1"
      }
    },
    "status": "DECLINED"
}
```


Tap-Initiated Sales for Debt Recovery with EMV Data

A sale transaction is a bundled authorization and capture. When a cardholder attempts to use a blocked card at the transit reader, create a fresh debt recovery sale request using the chip data from the new tap, along with the fare amount of the previous declined authorization.

Endpoint

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

Example: Mastercard Tap-Initiated Sale for Debt Recovery with EMV Data Using the REST API

Request


```
{
  "clientReferenceInformation": {
    "comments": "TransitDA Debt recovery tap sale",
    "code": "10000575MC",
    "transactionId": "20000575MC",
    "partner": {
      "thirdPartyCertificationNumber": "BPCDRC220403",
      "solutionId": "548UHQ8Z"
    }
  },
  "processingInformation": {
    "industryDataType": "transit",
    "commerceIndicator": "retail",
    "capture": "true",
    "captureOptions": {
      "dateToCapture": "0425"
    },
    "authorizationOptions": {
      "authIndicator": "1",
      "debtRecoveryIndicator": "true",
      "transportationMode": "00"
    }
  },
  "orderInformation": {
    "amountDetails": {
      "totalAmount": "10.00",
      "currency": "EUR"
    }
  },
  "paymentInformation": {
    "card": {
      "type": "002"
    },
    "initiationChannel": "00"
  },
  "pointOfSaleInformation": {
    "terminalId": "12345678",
    "catLevel": "2",
    "entryMode": "contactless",
    "terminalCapabiity": "5",
      "terminalPinCapability": "0",
    "emv": {
      "tags":
 "5F2A0209768407A0000000004101 09F360200039F03060000000000009C01005F3401019F10120110A0000F040000000000
    },
    "trackData": ";5413330089700042=49122010123456789?",
    "serviceCode": "201"
  }
}
```

Response to a Declined Request


```
{
  "_links": {
    "self": {
      "method": "GET",
      "href": "/pts/v2/payments/650887802747651300400"
    }
  },
  "clientReferenceInformation": {
    "code": "10000575MC",
    "partner": {
      "solutionId": "548UHQ8Z"
    },
    "transactionId": "20000575MC"
  },
  "errorInformation": {
    "reason": "PROCESSOR_DECLINED",
    "message": "Decline - General decline of the card. No other information provided by the issuing bank."
  },
  "id": "6508878027476513004 00",
  "pointOfSaleInformation": {
    "emv": {
      "tags":
"9F36020015910AB58D60185BEF0247303072179F18043030303186 0E04DA9F5809 03B1BAEDFD1438BA48"
    }
  },
  "processorInformation": {
    "systemTraceAuditNumber": "162956",
    "networktransactionId": "016153570198200",
    "retrievalReferenceNumber": "211511162956",
    "transactionId": "016153570198200",
    "responseCode": "05",
    "avs": {
      "code": "2"
    }
  },
  "status": "DECLINED"
}
```


Example: Tap-Initiated Sale for Debt Recovery with EMV Data Using the REST API

This is an example for all card types except Mastercard.

Request


```
{
  "clientReferenceInformation": {
    "comments": "TransitDA Debt recovery tap sale",
    "code": "10000575",
    "transactionId": "20000575",
    "partner": {
      "thirdPartyCertificationNumber": "BPCDRC220403",
      "solutionId": "548UHQ8Z"
```

```
      }
    },
    "processingInformation": {
      "industryDataType": "transit",
      "commerceIndicator": "retail",
      "capture": "true",
      "captureOptions": {
        "dateToCapture": "0425"
      },
      "authorizationOptions": {
        "authIndicator": "1",
        "debtRecoveryIndicator": "true",
        "deferredAuthIndicator": "true",
        "transportationMode": "00"
      }
    },
    "orderInformation": {
      "amountDetails": {
        "totalAmount": "10.00",
        "currency": "EUR"
      }
    },
    "paymentInformation": {
      "card": {
        "type": "001"
      },
      "initiationChannel": "00"
    },
    "pointOfSaleInformation": {
      "terminalId": "12345678",
      "catLevel": "2",
      "entryMode": "contactless",
      "terminalCapabiity": "5",
        "terminalPinCapability": "0",
      "emv": {
        "tags":
 "5F2A0209768407A0000000004101 09F360200039F03060000000000009C01005F3401019F10120110A0000F040000000000
      },
      "trackData": ";4413330089700042=49122010123456789?",
      "serviceCode": "201"
    }
}
```

Response to a Declined Request

```
{
    "_links": {
      "self": {
        "method": "GET",
        "href": "/pts/v2/payments/6508878027476513004004"
      }
    },
    "clientReferenceInformation": {
      "code": "10000575",
      "partner": {
        "solutionId": "548UHQ8Z"
```

```
      },
      "transactionId": "20000575"
   },
   "errorInformation": {
      "reason": "PROCESSOR_DECLINED",
      "message": "Decline - General decline of the card. No other information provided by the issuing bank."
   },
   "id": "6508878027476513004004",
   "pointOfSaleInformation": {
      "emv": {
         "tags":
 "9F36020015910AB58D60185BEF0247303072179F18043030303186 0E04DA9F5809 03B1BAEDFD1438BA48"
      }
   },
   "processorInformation": {
      "systemTraceAuditNumber": "162956",
      "networktransactionId": "016153570198200",
      "retrievalReferenceNumber": "211511162956",
      "transactionId": "016153570198200",
      "responseCode": "05",
      "avs": {
         "code": "2"
      }
   },
   "status": "DECLINED"
}
```

Merchant-Initiated Sales for Debt Recovery with Stored Card Data

Request a bundled authorization and capture to perform a merchant-initiated sale for debt recovery.

Endpoint

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

Example: Merchant-Initiated Sale for Debt Recovery with Stored Card Data

Request

```
{
   "clientReferenceInformation": {
      "comments": "TransitDA Debt recovery MIT sale",
      "code": "10000579",
      "transactionId": "20000579",
      "partner": {
         "thirdPartyCertificationNumber": "BPCDRC220403",
```

```
        "solutionId": "548UHQ8Z"
      }
    },
    "processingInformation": {
      "commerceIndicator": "moto",
      "industryDataType": "transit",
      "reconciliationId": "1111",
      "capture": "true",
      "authorizationOptions": {
        "debtRecoveryIndicator": "true",
        "authIndicator": "1",
        "ignoreAvsResult": "true",
        "ignoreCvResult": "true",
        "transportationMode": "01",
        "initiator": {
          "type": "merchant",
          "storedCredentialUsed": "true",
          "merchantInitiatedTransaction": {
            "reason": "1",
            "previoustransactionId": "016153570198200"
          }
        }
      }
    },
    "paymentInformation": {
      "card": {
        "number": "5413330089700042",
        "expirationMonth": "12",
        "expirationYear": "2049",
        "type": "002"
      }
    },
    "orderInformation": {
      "amountDetails": {
        "totalAmount": "10.00",
        "currency": "EUR"
      }
    }
}
```

Response to a Declined Request

```
{
    "_links": {
      "self": {
        "method": "GET",
        "href": "/pts/v2/payments/6508879024886569004002"
      }
    },
    "clientReferenceInformation": {
      "code": "10000579",
      "partner": {
        "solutionId": "548UHQ8Z"
      },
      "transactionId": "20000579"
    },
```

```
    "errorInformation": {
      "reason": "PROCESSOR_DECLINED",
      "message": "Decline - General decline of the card. No other information provided by the issuing bank."
    },
    "id": "6508879024886569004002",
    "pointOfSaleInformation": {
      "emv": {
        "tags":
 "9F36020015910AB58D60185BEF0247303072179F18043030303186 0E04DA9F5809 03B1BAEDFD1438BA48"
      }
    },
    "processorInformation": {
      "systemTraceAuditNumber": "162971",
      "networktransactionId": "016153570198200",
      "retrievalReferenceNumber": "211511162971",
      "transactionId": "016153570198200",
      "responseCode": "05",
      "avs": {
        "code": "1"
      }
    },
    "status": "DECLINED"
}
```


Stand-Alone Credits with Card Data

> **Warning**
> Follow these guidelines to prevent unauthorized credits.

A stand-alone credit is a credit that is not linked to a capture. There is no time limit for requesting a stand-alone credit.
When a request for a credit is successful, the issuing bank for the payment card takes money out of your merchant bank account and returns it to the customer. It usually takes two to four days for your acquiring bank to transfer funds from your merchant bank account.
Carefully control access to the credit service. Do not request this service directly from your customer interface. Instead, incorporate this service as part of your customer service process.

Endpoint

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

Example: Stand-Alone Credit with Card Data

Request


```
{
```

```
  "clientReferenceInformation": {
    "comments": "CREDIT Automatic or CREDIT Manual",
    "code": "10000599",
    "transactionId": "20000599",
    "partner": {
      "thirdPartyCertificationNumber": "BPCDRC220403",
      "solutionId": "548UHQ8Z"
    }
  },
  "paymentInformation": {
    "card": {
      "number": "4761731000000027",
      "expirationMonth": "12",
      "expirationYear": "2024",
      "type": "001"
    }
  },
  "orderInformation": {
    "amountDetails": {
      "totalAmount": "20.00",
      "currency": "EUR"
    }
  }
}
```

Response to a Successful Request

```
{
  "_links": {
    "void": {
      "method": "POST",
      "href": "/pts/v2/credits/650888404285658410400 4/voids"
    },
    "self": {
      "method": "GET",
      "href": "/pts/v2/credits/650888404285658410400 4"
    }
  },
  "clientReferenceInformation": {
    "code": "10000599",
    "partner": {
      "solutionId": "548UHQ8Z"
    },
    "transactionId": "20000599"
  },
  "creditAmountDetails": {
    "currency": "EUR",
    "creditAmount": "20.00"
  },
  "id": "6508884042856584104004",
  "orderInformation": {
    "amountDetails": {
      "currency": "EUR"
    }
  },
  "paymentAccountInformation": {
```

```
      "card": {
        "type": "001"
      }
    },
    "paymentInformation": {
      "tokenizedCard": {
        "type": "001"
      },
      "card": {
        "type": "001"
      }
    },
    "processorInformation": {
      "approvalCode": "831000",
      "networktransactionId": "016153570198200",
      "retrievalReferenceNumber": "211512164958",
      "responseCode": "00"
    },
    "reconciliationId": "65088840428565841040004",
    "status": "PENDING",
    "submitTimeUtc": "2022-04-25T12:06:44Z"
}
```


Capture an Authorization

When a transaction is below the threshold for First Ride Risk protection, use the capture service to capture funds from a declined authorization.

Endpoint

Production: `POST https://api.cybersource.com/pts/v2/payments/{id}/captures`
Test: `POST https://apitest.cybersource.com/pts/v2/payments/{id}/captures`
The `{id}` is the transaction ID returned in the authorization response.

Example: Capture an Authorization

Request


```
{
  "clientReferenceInformation": {
     "comments": "TransitDA BAU capture",
     "transactionId": "14987654321",
     "partner": {
       "thirdPartyCertificationNumber": "123456789012"
     }
  },
  "orderInformation": {
   "amountDetails": {
    "totalAmount": "10.00",
    "currency": "EUR"
   }
  }
```

```
}
```

Response to a Successful Request

```
{
  "_links": {
    "void": {
      "method": "POST",
      "href": "/pts/v2/captures/6484688186356910704004/voids"
    },
    "self": {
      "method": "GET",
      "href": "/pts/v2/captures/648468818635691070400 4"
    }
  },
  "clientReferenceInformation": {
    "comments": "capture",
    "code": "testcode1012",
    "transactionId": "14987654321"
  },
  "id": "6484688186356910704004",
  "orderInformation": {
    "amountDetails": {
      "totalAmount": "10.00",
      "currency": "EUR"
    }
  },
  "reconciliationId": "fgssgsgsgsfg",
  "status": "PENDING",
  "submitTimeUtc": "2022-03-28T12:00:18Z"
}
```

Authorization Reversals

Use the authorization reversal service to reverse an unnecessary or undesired authorization.

Endpoint

Production: `POST https://api.cybersource.com/pts/v2/payments/{id}/reversals`
Test: `POST https://apitest.cybersource.com/pts/v2/payments/{id}/reversals`
The `{id}` is the transaction ID returned in the authorization response.

Example: Reversing a Mass Transit Authorization

Request

```
{
  "clientReferenceInformation": {
    "comments": "REVERSAL Timeout",
    "transactionId": "11987654321",
    "partner": {
```

```
        "thirdPartyCertificationNumber": "123456789012"
      }
    },
    "reversalInformation": {
      "amountDetails": {
        "totalAmount": "300.00",
        "currency": "EUR"
      }
    }
}
```

Response to a Successful Request

```
{
    "_links": {
      "self": {
        "method": "GET",
        "href": "/pts/v2/reversals/6484678664766823004004"
      }
    },
    "clientReferenceInformation": {
      "code": "123456",
      "transactionId": "11987654321"
    },
    "id": "6484678664766823004004",
    "orderInformation": {
      "amountDetails": {
        "currency": "EUR"
      }
    },
    "processorInformation": {
      "responseDetails": "ABC",
      "responseCode": "00"
    },
    "reconciliationId": "6484678664766823004004",
    "reversalAmountDetails": {
      "reversedAmount": "300.00",
      "currency": "EUR"
    },
    "status": "REVERSED",
    "submitTimeUtc": "2022-03-28T11:44:26Z"
}
```

Timeout Reversal

Request a timeout reversal when an authorization is not completed within the time allowed and times out.

Production: POST https://api.cybersource.com/pts/v2/reversals
Test: POST https://apitest.cybersource.com/pts/v2/reversals

Example: Timeout Reversal

Request

```
{
  "clientReferenceInformation": {
    "comments": "REVERSAL Timeout",
    "transactionId": "78885555"
  },
  "reversalInformation": {
    "amountDetails": {
      "totalAmount": "10.00"
    },
    "reason": "testing"
  }
}
```

Response to a Successful Request

```
{
  "_links": {
    "self": {
      "method": "GET",
      "href": "/pts/v2/reversals/650285470710643110400 4"
    }
  },
  "clientReferenceInformation": {
    "code": "1650285470690",
    "transactionId": "78885555"
  },
  "id": "650285470710643110400 4",
  "orderInformation": {
    "amountDetails": {
      "currency": "EUR"
    }
  },
  "pointOfSaleInformation": {
    "emv": {
      "tags":
"5004564953419F26087C14E9BE1F1065094F07A0000000031010820220009F360203709F0702C0809F2701409F100706010A
    }
  },
  "processorInformation": {
    "responseCode": "00"
  },
  "reconciliationId": "650285470710643110400 4",
  "reversalAmountDetails": {
    "reversedAmount": "10.00",
    "currency": "EUR"
  },
  "status": "REVERSED",
  "submitTimeUtc": "2022-04-18T12:37:50Z"
}
```

Response to a Decline Request

```
{
 "id": "6502857670496139204005",
 "submitTimeUtc": "2022-04-18T12:42:47Z",
 "status": "INVALID_REQUEST",
 "reason": "INVALID_DATA",
 "message": "Declined - One or more fields in the request contains invalid data"
}
```

Timeout Void

Request a timeout void for a previous authorization, capture, refund, or credit when you do not receive a response within the time allowed and the transaction times out. To use this feature, you must include a unique value in the **clientReferenceInformation.transactionId** field in your payment, capture, refund, or credit request and use the same unique value for the **clientReferenceInformation.transactionId** field in this request to reverse the transaction.

Endpoint

Production: `POST https://api.cybersource.com/pts/v2/voids/`
Test: `POST https://apitest.cybersource.com/pts/v2/voids/`

Example: Timeout Void

Request

```
{
 "clientReferenceInformation": {
  "comments": "VOID Timeout",
  "transactionId": "888858556"
 }
}
```

Response to a Successful Request

```
{
 "_links": {
  "self": {
   "method": "GET",
   "href": "/pts/v2/voids/6502849034136438604002"
  }
 },
 "clientReferenceInformation": {
  "code": "1650284903396",
  "transactionId": "888858556"
 },
 "id": "6502849034136438604002",
 "orderInformation": {
  "amountDetails": {
```

```
    "currency": "EUR"
   }
  },
  "status": "VOIDED",
  "submitTimeUtc": "2022-04-18T12:28:23Z",
  "voidAmountDetails": {
   "currency": "EUR",
   "voidAmount": "10.00"
  }
}
```

Response to a Declined Request

```
{
  "id": "6502858209346457804004",
  "submitTimeUtc": "2022-04-18T12:43:41Z",
  "status": "INVALID_REQUEST",
  "reason": "INVALID_DATA",
  "message": "Declined - One or more fields in the request contains invalid data"
}
```

Debit and Prepaid Card Processing

This section shows you how to process authorizations that use a debit or prepaid card.

Requirements

In Canada, to process domestic debit transactions on Visa Platform Connect with Mastercard, you must contact customer support to have your account configured for this feature.

Related Information

- See *Debit and Prepaid Card Payments* on page 31 for a description of the debit or prepaid card transactions you can process.

Additional Resources for Debit and Prepaid Payments

For more information, see these guides:

- *PIN Debit Processing in Card Present Connect | Retail Integration Guide*
- *API field reference guide for the REST API*
- Github repositories: *https://github.com/Cybersource?q=rest-sample&type=all&language=&sort=*

Processing Debit and Prepaid Authorizations

This section shows you how to process an authorization using debit and prepaid cards.

Endpoint

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

Required Fields for Processing Debit and Prepaid Authorizations

Use these required fields for processing debit and prepaid authorizations.

> **Important**
>
> When using relaxed requirements for address data and the expiration date, not all fields in this list are required. It is your responsibility to determine whether your account is enabled to use this feature and which fields are required. For details about relaxed requirements, see *Relaxed Requirements for Address Data and Expiration Date in Payment Transactions* on page 264.

clientReferenceInformation.code

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.type

paymentInformation.card.expirationMonth

paymentInformation.card.expirationYear

paymentInformation.card.number

Related Information

- *API field reference guide for the REST API*

Country Specific Required Fields to Process Debit and Prepaid Authorizations

Use these country-specific required fields to process a debit or prepaid authorization.

Argentina

merchantInformation.taxId	**Required for Mastercard transactions.**
merchantInformation.transactionLocalDateTime	**Required in Argentina when the time zone is not included in your account. Otherwise, this field is optional.**

Brazil

paymentInformation.card.sourceAccountType	**Required for combo card transactions.**
paymentInformation.card.sourceAccountTypeRequired	**Required for combo card line-of-credit and prepaid-card transactions.**

Chile

merchantInformation.taxId	**Required for Mastercard transactions.**

Paraguay

merchantInformation.taxId	**Required for Mastercard transactions.**

Saudi Arabia

processingInformation.authorizationOptions.transactionMode

Taiwan

paymentInformation.card.hashedNumber

Related Information

- *API field reference guide for the REST API*

Optional Field for Processing Debit and Prepaid Authorizations

You can use this optional field to include additional information when processing debit and prepaid authorizations.

processingInformation.linkId	**Set this field to the request ID that was returned in the response message from the original authorization request.**

Related Information

- *API field reference guide for the REST API*

REST Example: Processing Debit and Prepaid Authorizations

Endpoint:

- Production: POST https://api.cybersource.com/pts/v2/payments
- Test: POST https://apitest.cybersource.com/pts/v2/payments

Request

```
{
 "orderInformation" : {
  "billTo" : {
   "country" : "US",
   "firstName" : "John",
   "lastName" : "Deo",
   "address1" : "901 Metro Center Blvd",
   "postalCode" : "40500",
   "locality" : "Foster City",
   "administrativeArea" : "CA",
   "email" : "test@cybs.com"
},
  "amountDetails" : {
   "totalAmount" : "100.00",
   "currency" : "USD"
  }
 },
 "paymentInformation" : {
  "card" : {
   "expirationYear" : "2031",
   "number" : "4111111111111111",
   "securityCode" : "123",
   "expirationMonth" : "12",
   "type" : "001"
  }
 }
}
```

Response to a Successful Request

```
{
 "_links" : {
  "authReversal" : {
   "method" : "POST",
   "href" : "/pts/v2/payments/6595482584316313203494/reversals"
  },
  "self" : {
   "method" : "GET",
   "href" : "/pts/v2/payments/659548258431631320 3494"
  },
  "capture" : {
   "method" : "POST",
   "href" : "/pts/v2/payments/6595482584316313203494/captures"
  }
 },
 "clientReferenceInformation" : {
  "code" : "RTS-Auth"
```

```
  },
  "consumerAuthenticationInformation" : {
    "token" : "Axj/7wSTZYq1MhJBMfMmAEQs2auWrRwyauGjNi2ZsWbJgzaOWiaVA+JbK
          AU0qB8S2VpA6cQIp4ZNvG2YbC9eM4E5NlirUyEkEx8yYAAA4A1c"
  },
  "id" : "6595482584316313203494",
  "orderInformation" : {
    "amountDetails" : {
      "authorizedAmount" : "100.00",
      "currency" : "USD"
    }
  },
  "paymentAccountInformation" : {
    "card" : {
      "type" : "001"
    }
  },
  "paymentInformation" : {
    "tokenizedCard" : {
      "type" : "001"
    },
    "card" : {
      "type" : "001"
    }
  },
  "processorInformation" : {
    "systemTraceAuditNumber" : "853428",
    "approvalCode" : "831000",
    "cardVerification" : {
      "resultCodeRaw" : "M",
      "resultCode" : "M"
    },
    "merchantAdvice" : {
      "code" : "01",
      "codeRaw" : "M001"
    },
    "responseDetails" : "ABC",
    "networkTransactionId" : "016153570198200",
    "retrievalReferenceNumber" : "221517853428",
    "consumerAuthenticationResponse" : {
      "code" : "2",
      "codeRaw" : "2"
    },
    "transactionId" : "016153570198200",
    "responseCode" : "00",
    "avs" : {
      "code" : "Y",
      "codeRaw" : "Y"
    }
  }
}
```

Enabling Debit and Prepaid Partial Authorizations

Partial authorizations and balance responses are special features that are available for debit cards and prepaid cards. This section shows you how to enable partial authorizations for a specific transaction.
To globally process domestic debit transactions on Visa Platform Connect with Mastercard in Canada, you must contact customer support to have your account configured for this feature.

Field Specific to this Use Case

Include this field in addition to the fields required for a standard authorization request:

- Indicate that this request is a partial authorization.
  Set the **processingInformation.authorizationOptions.partialAuthIndicator** to `true`.

Endpoint

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

Required Fields for Enabling Debit and Prepaid Partial Authorizations

Use these required fields for enabling debit and prepaid partial authorizations.

Important

When using relaxed requirements for address data and the expiration date, not all fields in this list are required. It is your responsibility to determine whether your account is enabled to use this feature and which fields are required. For details about relaxed requirements, see *Relaxed Requirements for Address Data and Expiration Date in Payment Transactions* on page 264.

clientReferenceInformation.code

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.type

paymentInformation.card.expirationMonth

paymentInformation.card.expirationYear

paymentInformation.card.number

processingInformation.authorizationOptions.partialAuthIndicator

Set the value to `true`.

Related Information

- *API field reference guide for the REST API*

Optional Field for Enabling Debit and Prepaid Partial Authorizations

You can use these optional fields to include additional information when enabling debit and prepaid partial authorizations.

processingInformation.linkId

Set this field to the request ID that was returned in the response message from the original authorization request.

Related Information

- *API field reference guide for the REST API*

REST Example: Enabling Debit and Prepaid Partial Authorizations

Endpoint:

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

Request

```
{
  "clientReferenceInformation" : {
    "code" : "TC50171_3"
  },
  "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"
   },
   "amountDetails" : {
    "totalAmount" : "1000.00",
    "currency" : "USD"
   }
  },
  "paymentInformation" : {
   "card" : {
    "expirationYear" : "2031",
    "number" : "5555555555554444",
    "securityCode" : "123",
    "expirationMonth" : "12",
    "type" : "002"
   }
  },
 "processingInformation" : {
  "authorizationOptions" : {
    "partialAuthIndicator" : "true"
   }
  }
}
```

Response to a Successful Request

```
{
  "_links" : {
   "self" : {
    "method" : "GET",
    "href" : "/pts/v2/payments/6595549144566655003494"
   }
  },
  "clientReferenceInformation" : {
   "code" : "TC50171_3"
  },
  "id" : "6595549144566655003494",
  "orderInformation" : {
   "amountDetails" : {
    "totalAmount" : "1000.00",
    "authorizedAmount" : "499.01",
    "currency" : "USD"
   }
  },
  "paymentInformation" : {
   "accountFeatures" : {
    "currency" : "usd",
    "balanceAmount" : "0.00"
   }
  },
  "pointOfSaleInformation" : {
   "terminalId" : "261996"
```

```
  },
  "processorInformation" : {
    "merchantNumber" : "000000092345678",
    "approvalCode" : "888888",
    "cardVerification" : {
      "resultCode" : ""
    },
    "networkTransactionId" : "123456789619999",
    "transactionId" : "123456789619999",
    "responseCode" : "100",
    "avs" : {
      "code" : "X",
      "codeRaw" : "I1"
    }
  },
  "reconciliationId" : "56059417N6C86KTJ",
  "status" : "PARTIAL_AUTHORIZED",
  "submitTimeUtc" : "2022-08-03T19:28:34Z"
}
```


Disabling Debit and Prepaid Partial Authorizations

This topic shows you how to successfully disable partial authorizations for specific transactions.

Field Specific to this Use Case

Include this field in addition to the fields required for a standard authorization request:

- Indicate that this request is not a partial authorization.
  Set the `processingInformation.authorizationOptions.partialAuthIndicator` to `false`.

Endpoint

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

Required Field for Disabling Debit and Prepaid Partial Authorizations

Use these required fields for disabling debit and prepaid partial authorizations.

> **Important**
>
> When using relaxed requirements for address data and the expiration date, not all fields in this list are required. It is your responsibility to determine whether your account is enabled to use this feature and which fields are required. For details

about relaxed requirements, see *Relaxed Requirements for Address Data and Expiration Date in Payment Transactions* on page 264.

clientReferenceInformation.code

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.type

paymentInformation.card.expirationMonth

paymentInformation.card.expirationYear

paymentInformation.card.number

processingInformation.authorizationOptions.partialAuthIndicator

Set the value to `false` in an authorization or sale request. When you do so, only that specific transaction is disabled for partial authorization.

Related Information

- *API field reference guide for the REST API*

Optional Field for Disabling Debit and Prepaid Partial Authorizations

You can use this optional field to include additional information when disabling debit and prepaid partial authorizations.

processingInformation.linkId

Set this field to the request ID that was returned in the response message from the original authorization request.

Related Information

- *API field reference guide for the REST API*

REST Example: Disabling Debit and Prepaid Partial Authorizations

Endpoint:

- Production: POST https://api.cybersource.com/pts/v2/payments
- Test: POST https://apitest.cybersource.com/pts/v2/payments

Request

```
{
   "processingInformation":{
  "authorizationOptions":{
   "partialAuthIndicator": "false"
  }
 },
  "clientReferenceInformation" : {
    "code" : "TC50171_3"
  },
  "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"
    },
    "amountDetails" : {
      "totalAmount" : "501.00",
      "currency" : "USD"
    }
  },
  "paymentInformation" : {
    "card" : {
      "expirationYear" : "2031",
      "number" : "5555555555554444",
      "securityCode" : "123",
      "expirationMonth" : "12",
      "type" : "002"
    }
  }
}
```

Response to a Successful Request

```
{
   "_links": {
      "self": {
         "method": "GET",
```

```
        "href": "/pts/v2/payments/6595545423896900104953"
      }
    },
    "clientReferenceInformation": {
      "code": "TC50171_3"
    },
    "errorInformation": {
      "reason": "PROCESSOR_DECLINED",
      "message": "Decline - General decline of the card.
            No other information provided by the issuing bank."
    },
    "id": "6595545423896900104953",
    "pointOfSaleInformation": {
      "terminalId": "111111"
    },
    "processorInformation": {
      "networkTransactionId": "123456789619999",
      "transactionId": "123456789619999",
      "responseCode": "100",
      "avs": {
        "code": "X",
        "codeRaw": "I1"
      }
    },
    "status": "DECLINED"
}
```

Airline Data Processing

This section describes how to process airline payments.

Requirement

When you are ready to go live with airline data processing, contact Cybersource Customer Support to have your account configured to process airline data. If your account is not enabled, and you try to send airline transactions, you will receive an error for invalid data.

Related Information

- See *Airline Data* on page 32 for information about and requirements for processing payments that include airline data.
- See *Airline Data Reference Information* on page 35 for a list and description of the different document type and ancillary service category codes that are used when processing payments that include airline data.

Additional Resources for Airline Data

For more information, see these guides:

- *Card Present Connect | Retail Guide*
- *API field reference guide for the REST API*
- Github repositories: *https://github.com/Cybersource?q=rest-sample&type=all&language=&sort=*

Airline Travel Legs

Some processors require travel legs in the API service request. This section describes how to successfully include travel legs in an API request.

Using Travel Legs

To include travel legs in an airline transaction, include one or more travel legs in the **legs[]** array.
For example, these three travel legs are valid:

```
"travelInformation": {
  "transit": {
   "airline": {
    "legs": [
     {
      "carrierCode": "XX"
     },
     {
      "carrierCode": "XZ"
     },
     {
      "carrierCode": "XX"
     }
    ]
   }
```

Important

If you skip a number, Cybersource ignores the legs that follow the skipped number.

Travel Leg Limitations

Some processors limit the number of travel legs for each trip based on the card type.

Authorizations

This section describes how to process an airline authorization.

Authorization Restrictions

Ticket purchases that include multiple passengers may be included in a single authorization request, but you must make separate capture requests for every passenger.
If any ancillary purchases are made at the same time as the ticket purchase, you may include all items in a single authorization request, but you must separate the ancillary and ticket purchases into their own capture requests.
If any ancillary purchases are made not at the same time as the ticket purchase, you must make separate authorization and capture requests for the ancillary and ticket purchases.

Endpoint

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

Required Fields for Authorizing an Airline Payment

Include these required fields for authorizing an airline payment.

> **Important**
>
> When using relaxed requirements for address data and the expiration date, not all fields in this list are required. It is your responsibility to determine whether your account is enabled to use this feature and which fields are required. For details about relaxed requirements, see *Relaxed Requirements for Address Data and Expiration Date in Payment Transactions* on page 264.

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

processingInformation.industryDataType **Set the value to `airline`.**

Related Information

- *API field reference guide for the REST API*

REST Example: Authorizing an Airline Payment

Use this example as a reference for authorizing an airline payment.

Request

```
{
  "processingInformation": {
    "industryDataType": "airline"
  },
  "paymentInformation": {
    "card": {
      "number": "4111111111111111",
```

```
    "expirationMonth": "12",
    "expirationYear": "2031"
   }
  },
  "orderInformation": {
   "amountDetails": {
    "totalAmount": "500.00",
    "currency": "usd"
   },
   "billTo": {
    "firstName": "John",
    "lastName": "Doe",
    "address1": "123 Happy St.",
    "locality": "Sunny Town",
    "administrativeArea": "CA",
    "postalCode": "12345-1234",
    "country": "US",
    "email": "test@cybs.com"
   }
  }
}
```

Response to a Successful Request

```
{
  "_links": {
   "authReversal": {
    "method": "POST",
    "href": "/pts/v2/payments/6823009451126309503954/reversals"
   },
   "self": {
    "method": "GET",
    "href": "/pts/v2/payments/682300945112630950 3954"
   },
   "capture": {
    "method": "POST",
    "href": "/pts/v2/payments/6823009451126309503954/captures"
   }
  },
  "clientReferenceInformation": {
   "code": "1682300945230"
  },
  "id": "6823009451126309503954",
  "orderInformation": {
   "amountDetails": {
    "authorizedAmount": "500.00",
    "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": "67720603YGMSE5JE",
  "status": "AUTHORIZED",
  "submitTimeUtc": "2023-04-24T01:49:05Z"
}
```


Captures for Ticket Purchases

This section shows the fields necessary to capture an airline payment for ticket purchases.
Captures for ticket purchases must be made separately from captures for ancillary purchases. For more information about how to capture an ancillary purchase, see *Captures for Ancillary Purchases* on page 181.

Travel Legs

You can use travel leg fields for trips that have multiple legs. For more information on how to use travel leg fields, see *Airline Data* on page 32.

Leg Limitations

Visa Platform Connect limits the maximum number of legs for each trip based on card type. This table shows the maximum number of legs for each trip based on card type.

Visa Platform Connect Leg Limitations

Supported Card Types	Maximum Number of Trip Legs
American Express	4
Discover	4
Mastercard	4

Supported Card Types	Maximum Number of Trip Legs
Visa	4

Endpoint

Production: `POST https://api.cybersource.com/pts/v2/payments/{id}/captures`
Test: `POST https://apitest.cybersource.com/pts/v2/payments/{id}/captures`
The `{id}` is the transaction ID returned in the authorization response.

Required Fields for Capturing an Airline Payment

Include these required fields to capture an airline payment for ticket purchases.

orderInformation.amountDetails.currency

orderInformation.amountDetails.totalAmount

processingInformation.industryDataType **Set the value to `airline`.**

Related Information

- *API field reference guide for the REST API*

Card-Specific Field to Capture an Airline Payment

This section includes card-specific information.

Mastercard

Use this card-specific field in addition to the required fields when capturing an authorization with a Mastercard.

travelInformation.transit.airline.ticketIssuer.code

Related Information

- *API field reference guide for the REST API*

Optional Fields for Capturing an Airline Payment

You can use these optional fields to include additional information when capturing an airline payment.

orderInformation.amountDetails.taxAmount

orderInformation.amountDetails.taxDetails[].amount

orderInformation.lineItems[].taxDetails[].code

orderInformation.lineItems[].totalAmount

travelInformation.agency.code

travelInformation.agency.name

travelInformation.transit.airline.customerCode

travelInformation.transit.airline.documentType **For a list of possible values, see *Airline Document Type Codes* on page 35.**

travelInformation.transit.airline.exchangeTicketFeeAmount

travelInformation.transit.airline.legs[].conjunctionTicket

travelInformation.transit.airline.legs[].couponNumber

travelInformation.transit.airline.legs[].endorsementsRestrictions

travelInformation.transit.airline.legs[].exchangeTicketNumber

travelInformation.transit.airline.legs[].fareBasis

travelInformation.transit.airline.legs[].feeAmount

travelInformation.transit.airline.legs[].stopoverIndicator

travelInformation.transit.airline.legs[].taxAmount

travelInformation.transit.airline.planNumber

travelInformation.transit.airline.ticketChangeIndicator

travelInformation.transit.airline.ticketIssueDate

travelInformation.transit.airline.totalFeeAmount

travelInformation.transit[].exchangeTicketAmount

Related Information

- *API field reference guide for the REST API*

REST Example: Capturing an Airline Payment

Endpoint:

- Production: POST https://api.cybersource.com/pts/v2/payments/{id}/captures
- Test: POST https://apitest.cybersource.com/pts/v2/payments/{id}/captures

Use this example as a reference for capturing an airline payment.

Request

```
{
 "clientReferenceInformation": {
  "code": "TC50171_3"
 },
 "processingInformation": {
  "industryDataType": "airline"
 },
 "orderInformation": {
  "amountDetails": {
```

```
    "totalAmount": "500.00",
    "currency": "USD"
   }
  }
}
```

Response to a Successful Request

```
{
 "_links": {
  "void": {
   "method": "POST",
   "href": "/pts/v2/captures/6823025890736075903954/voids"
  },
  "self": {
   "method": "GET",
   "href": "/pts/v2/captures/6823025890736075903954"
  }
 },
 "clientReferenceInformation": {
  "code": "TC50171_3"
 },
 "id": "6823025890736075903954",
 "orderInformation": {
  "amountDetails": {
   "totalAmount": "500.00",
   "currency": "USD"
  }
 },
 "reconciliationId": "67720603YGMSE5JE",
 "status": "PENDING",
 "submitTimeUtc": "2023-04-24T02:16:29Z"
}
```

Captures for Ancillary Purchases

This section shows the fields necessary to capture an airline payment for ancillary purchases.
Ancillary purchases are any additional services, such as baggage, meals, and paid seats, that your customers can purchase. Captures for ancillary purchases must be made separately from captures for ticket purchases.

Endpoint

Production: `POST https://api.cybersource.com/pts/v2/payments/{id}/captures`
Test: `POST https://apitest.cybersource.com/pts/v2/payments/{id}/captures`
The `{id}` is the transaction ID returned in the authorization response.

Required Fields for Capturing an Authorization for Ancillary Purchases

Include these required fields to capture an airline payment for ancillary purchases.

orderInformation.amountDetails.currency

orderInformation.amountDetails.totalAmount

processingInformation.industryDataType **Set the value to `airline`.**

Related Information

- *API field reference guide for the REST API*

Ancillary Fields for Capturing an Authorization for an Ancillary Purchase

Choose from these optional ancillary fields to add additional information when capturing an ancillary purchase.

travelInformation.transit.airline.ancillaryInformation.connectedTicketNumber

travelInformation.transit.airline.ancillaryInformation.creditReasonIndicator

travelInformation.transit.airline.ancillaryInformation.passengerName

travelInformation.transit.airline.ancillaryInformation.service[].categoryCode **For a list of possible values, see** *Ancillary Service Category Codes* **on page 38.**

travelInformation.transit.airline.ancillaryInformation.service[].subCategoryCode

travelInformation.transit.airline.ancillaryInformation.ticketNumber

Related Information

- *API field reference guide for the REST API*

REST Example: Capturing an Authorization for an Ancillary Purchase

Endpoint:

- Production: `POST https://api.cybersource.com/pts/v2/payments/{id}/captures`
- Test: `POST https://apitest.cybersource.com/pts/v2/payments/{id}/captures`

Use this example as a reference for capturing an ancillary purchase with the ancillary fields.

Request

```
{
  "clientReferenceInformation": {
    "code": "TC50171_3"
  },
```

```
  "processingInformation": {
    "industryDataType": "airline"
  },
  "orderInformation": {
    "amountDetails": {
      "totalAmount": "500.00",
      "currency": "USD"
    }
  },
  "travelInformation": {
    "transit": {
      "airline": {
        "ancillaryInformation": {
          "ticketNumber": "123456789123456",
          "passengerName": "John Doe",
          "connectedTicketNumber": "654321987654321"
        }
      }
    }
  }
}
```

Response to a Successful Request

```
{
  "_links": {
    "void": {
      "method": "POST",
      "href": "/pts/v2/captures/6823030661646093703954/voids"
    },
    "self": {
      "method": "GET",
      "href": "/pts/v2/captures/6823030661646093703954"
    }
  },
  "clientReferenceInformation": {
    "code": "TC50171_3"
  },
  "id": "6823030661646093703954",
  "orderInformation": {
    "amountDetails": {
      "totalAmount": "500.00",
      "currency": "USD"
    }
  },
  "reconciliationId": "67221841NGMV8WOT",
  "status": "PENDING",
  "submitTimeUtc": "2023-04-24T02:24:26Z"
}
```

Refunds

This topic describes how to process an airline refund.

This service returns the funds used in the initial capture and requires the original capture ID.

Endpoint

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

Required Fields for Processing an Airline Refund

Include these required fields to process an airline refund.

orderInformation.amountDetails.currency

orderInformation.amountDetails.totalAmount

Related Information

- *API field reference guide for the REST API*

Optional Fields for Processing an Airline Refund

This section includes these types of optional fields for an airline refund:

- *General Optional Fields*
- *Optional Fields for Ticket Purchases*
- *Optional Fields for Ancillary Purchases*

General Optional Fields

You can use these optional fields to include additional information in any airline purchase.

travelInformation.agency.code

travelInformation.agency.name

travelInformation.transit.airline.arrivalDate

travelInformation.transit.airline.carrierName

travelInformation.transit.airline.clearingCount

travelInformation.transit.airline.clearingSequence

travelInformation.transit.airline.creditReasonIndicator

travelInformation.transit.airline.customerCode

travelInformation.transit.airline.documentType **For a list of possible values, see *Airline Document Type Codes* on page 35.**

travelInformation.transit.airline.electronicTicketIndicator

travelInformation.transit.airline.exchangeTicketFeeAmount

travelInformation.transit.airline.numberOfPassengers

travelInformation.transit.airline.passengerName

travelInformation.transit.airline.planNumber

travelInformation.transit.airline.purchaseType

travelInformation.transit.airline.reservationSystemCode

travelInformation.transit.airline.restrictedTicketDesciption

travelInformation.transit.airline.restrictedTicketIndicator

travelInformation.transit.airline.ticketChangeIndicator

travelInformation.transit.airline.ticketIssueDate

travelInformation.transit.airline.ticketIssuer.locality

travelInformation.transit.airline.ticketNumber

travelInformation.transit.airline.totalClearingAmount

travelInformation.transit.airline.totalFeeAmount

travelInformation.transit[].exchangeTicketAmount

Airline Optional Fields for Ticket Purchases

You can use these optional fields to include additional information when requesting an airline credit for a ticket purchase.

travelInformation.transit.airline.legs[].arrivalTime

travelInformation.transit.airline.legs[].arrivalTimeMeridian

travelInformation.transit.airline.legs[].carrierCode

travelInformation.transit.airline.legs[].class

travelInformation.transit.airline.legs[].conjunctionTicket

travelInformation.transit.airline.legs[].couponNumber

travelInformation.transit.airline.legs[].departureDate

travelInformation.transit.airline.legs[].departureTime

travelInformation.transit.airline.legs[].departureTimeMeridian

travelInformation.transit.airline.legs[].destinationAirportCode

travelInformation.transit.airline.legs[].endorsementsRestrictions

travelInformation.transit.airline.legs[].exchangeTicketNumber

travelInformation.transit.airline.legs[].fareBasis

travelInformation.transit.airline.legs[].feeAmount

travelInformation.transit.airline.legs[].flightNumber

travelInformation.transit.airline.legs[].originatingAirportCode

travelInformation.transit.airline.legs[].stopoverIndicator

travelInformation.transit.airline.legs[].taxAmount

travelInformation.transit.airline.legs[].totalFareAmount

Ancillary Optional Fields

You can use these optional fields to include additional information when requesting an airline credit for an ancillary purchase.

travelInformation.transit.airline.ancillaryInformation.connectedTicketNumber

travelInformation.transit.airline.ancillaryInformation.creditReasonIndicator

travelInformation.transit.airline.ancillaryInformation.passengerName

travelInformation.transit.airline.ancillaryInformation.service[].categoryCode For a list of possible values, see *Ancillary Service Category Codes* **on page 38.**

travelInformation.transit.airline.ancillaryInformation.service[].subCategoryCode

travelInformation.transit.airline.ancillaryInformation.ticketNumber

Related Information

- *API field reference guide for the REST API*

REST Example: Processing an Airline Refund

Endpoint:

- Production: POST https://api.cybersource.com/pts/v2/credits/
- Test: POST https://apitest.cybersource.com/pts/v2/credits/

Use this example as a reference for processing an airline refund.

Request

```
{
 "clientReferenceInformation": {
  "code": "TC50171_3"
 },
 "processingInformation": {
  "industryDataType": "airline"
 },
 "orderInformation": {
  "amountDetails": {
   "totalAmount": "500",
   "currency": "USD"
  }
 }
}
```

Response to a Successful Request


```
{
  "_links": {
    "void": {
      "method": "POST",
      "href": "/pts/v2/refunds/6823038625416445403955/voids"
    },
    "self": {
      "method": "GET",
      "href": "/pts/v2/refunds/6823038625416445403955"
    }
  },
  "clientReferenceInformation": {
    "code": "TC50171_3"
  },
  "id": "6823038625416445403955",
  "orderInformation": {
    "amountDetails": {
      "currency": "USD"
    }
  },
  "processorInformation": {
    "approvalCode": "888888",
    "responseCode": "100"
  },
  "reconciliationId": "67722608EGMV6Q7V",
  "refundAmountDetails": {
    "currency": "USD",
    "refundAmount": "500.00"
  },
  "status": "PENDING",
  "submitTimeUtc": "2023-04-24T02:37:42Z"
}
```


Credits

This topic describes how to process an airline credit.
This service distributes funds without requiring a capture ID.

Important

All fields used in the original transaction must be included in your request.

Endpoint

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

Required Fields for Processing an Airline Credit

Include these required fields to process an airline credit.

> **Important**
>
> When using relaxed requirements for address data and the expiration date, not all fields in this list are required. It is your responsibility to determine whether your account is enabled to use this feature and which fields are required. For details about relaxed requirements, see *Relaxed Requirements for Address Data and Expiration Date in Payment Transactions* on page 264.

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

processingInformation.industryDataType **Set the value to `airline`.**

Related Information

- *API field reference guide for the REST API*

Optional Fields for Processing an Airline Credit

This section includes these types of optional fields for an airline credit:

- *General Optional Fields*
- *Optional Fields for Ticket Purchases*
- *Optional Fields for Ancillary Purchases*

General Optional Fields

You can use these optional fields to include additional information in any airline purchase.

travelInformation.agency.code

travelInformation.agency.name

travelInformation.transit.airline.arrivalDate

travelInformation.transit.airline.carrierName

travelInformation.transit.airline.clearingCount

travelInformation.transit.airline.clearingSequence

travelInformation.transit.airline.creditReasonIndicator

travelInformation.transit.airline.customerCode

travelInformation.transit.airline.documentType For a list of possible values, see *Airline Document Type Codes* on page 35.

travelInformation.transit.airline.electronicTicketIndicator

travelInformation.transit.airline.exchangeTicketFeeAmount

travelInformation.transit.airline.numberOfPassengers

travelInformation.transit.airline.passengerName

travelInformation.transit.airline.planNumber

travelInformation.transit.airline.purchaseType

travelInformation.transit.airline.reservationSystemCode

travelInformation.transit.airline.restrictedTicketDesciption

travelInformation.transit.airline.restrictedTicketIndicator

travelInformation.transit.airline.ticketChangeIndicator

travelInformation.transit.airline.ticketIssueDate

travelInformation.transit.airline.ticketIssuer.locality

travelInformation.transit.airline.ticketNumber

travelInformation.transit.airline.totalClearingAmount

travelInformation.transit.airline.totalFeeAmount

travelInformation.transit[].exchangeTicketAmount

Airline Optional Fields for Ticket Purchases

You can use these optional fields to include additional information when requesting an airline credit for a ticket purchase.

travelInformation.transit.airline.legs[].arrivalTime

travelInformation.transit.airline.legs[].arrivalTimeMeridian

travelInformation.transit.airline.legs[].carrierCode

travelInformation.transit.airline.legs[].class

travelInformation.transit.airline.legs[].conjunctionTicket

travelInformation.transit.airline.legs[].couponNumber

travelInformation.transit.airline.legs[].departureDate

travelInformation.transit.airline.legs[].departureTime

travelInformation.transit.airline.legs[].departureTimeMeridian

travelInformation.transit.airline.legs[].destinationAirportCode

travelInformation.transit.airline.legs[].endorsementsRestrictions

travelInformation.transit.airline.legs[].exchangeTicketNumber

travelInformation.transit.airline.legs[].fareBasis

travelInformation.transit.airline.legs[].feeAmount

travelInformation.transit.airline.legs[].flightNumber

travelInformation.transit.airline.legs[].originatingAirportCode

travelInformation.transit.airline.legs[].stopoverIndicator

travelInformation.transit.airline.legs[].taxAmount

travelInformation.transit.airline.legs[].totalFareAmount

Ancillary Optional Fields

You can use these optional fields to include additional information when requesting an airline credit for an ancillary purchase.

travelInformation.transit.airline.ancillaryInformation.connectedTicketNumber

travelInformation.transit.airline.ancillaryInformation.creditReasonIndicator

travelInformation.transit.airline.ancillaryInformation.passengerName

travelInformation.transit.airline.ancillaryInformation.service[].categoryCode For a list of possible values, see *Ancillary Service Category Codes* **on page 38.**

travelInformation.transit.airline.ancillaryInformation.service[].subCategoryCode

travelInformation.transit.airline.ancillaryInformation.ticketNumber

Related Information

- *API field reference guide for the REST API*

REST Example: Processing an Airline Credit

Endpoint:

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

Use this example as a reference for crediting an airline payment.

Request

```
{
  "paymentInformation": {
    "card": {
      "number": "4111111111111111",
      "expirationMonth": "12",
      "expirationYear": "31"
    }
  },
  "orderInformation": {
    "amountDetails": {
      "totalAmount": "500.00",
      "currency": "USD"
    },
    "billTo": {
      "firstName": "John",
      "lastName": "Doe",
      "address1": "123 Happy St.",
      "locality": "Sunnyville",
      "administrativeArea": "CA",
      "postalCode": "12345",
      "country": "US",
      "email": "johndoe@test.com"
    }
  }
}
```

Response to a Successful Request

```
{
  "_links": {
    "void": {
      "method": "POST",
      "href": "/pts/v2/credits/6823065885666134104951/voids"
    },
    "self": {
      "method": "GET",
      "href": "/pts/v2/credits/6823065885666134104951"
    }
  },
  "clientReferenceInformation": {
    "code": "1682306588644"
  },
  "creditAmountDetails": {
    "currency": "USD",
    "creditAmount": "500.00"
  },
  "id": "6823065885666134104951",
  "orderInformation": {
    "amountDetails": {
      "currency": "USD"
    }
  },
```

```
  "paymentAccountInformation": {
    "card": {
      "type": "001"
    }
  },
  "paymentInformation": {
    "tokenizedCard": {
      "type": "001"
    },
    "card": {
      "type": "001"
    }
  },
  "processorInformation": {
    "approvalCode": "888888",
    "responseCode": "100"
  },
  "reconciliationId": "74259417PGM9TXHT",
  "status": "PENDING",
  "submitTimeUtc": "2023-04-24T03:23:08Z"
}
```

Japanese Payment Options Processing

This section shows you how to process an authorization with Japanese payment options (JPO).
JPO supports these payment methods:

- Single payment
- Bonus payment
- Installment payment
- Revolving payment
- Combination of bonus payment and installment payment

Requirements

- You have signed a contract with your acquirer.
- You have contacted your account provider for details about contracts and funding cycles. The funding cycle could differ when using JPO.
- Card holders who want to use JPO have signed a contract with an issuing bank.
- You have confirmed payment option availability with your account provider and card holder before implementing one of these payment options.

Related Information

- See *Japanese Payment Options* on page 40 for a description of JPO payments.

Authorize a Single Payment with Japanese Payment Options

This section shows you how to process an authorization of a single payment with Japanese Payment Options (JPO).

Limitations

- The only supported acquirer is Sumitomo Mitsui Card Co.
- The payment must use a Visa payment card issued in Japan, and the only supported acquirer is Sumitomo Mitsui Card Co.

Prerequisites

- You have signed a contract with your acquirer.
- You have contacted your account provider for details about contracts and funding cycles. The funding cycle could differ when using JPO.
- Card holders who want to use JPO have signed a contract with an issuing bank.
- You have confirmed payment option availability with your account provider and card holder before implementing one of these payment options.

Endpoint

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

Required Fields for Authorizing a Single Payment Using the JPO Method

Use these required fields for authorizing a single payment using the JPO method.

Important

When using relaxed requirements for address data and the expiration date, not all fields in this list are required. It is your responsibility to determine whether your account is enabled to use this feature and which fields are required. For details about relaxed requirements, see *Relaxed Requirements for Address Data and Expiration Date in Payment Transactions* on page 264.

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.japanPaymentOptions.businessName	**Business name in kanji characters.**
processingInformation.japanPaymentOptions.businessNameAlphaNumeric	
processingInformation.japanPaymentOptions.businessNameKatakana	
processingInformation.japanPaymentOptions.terminalID	**Required for card-present transactions. Unique Japan Credit Card Association (JCCA) terminal identifier that is provided by Cybersource.**

Related Information

- *API field reference guide for the REST API*

REST Example: Authorizing a JPO Single Payment

This example shows an authorization request for a card-not-present single payment that uses Japanese payment options (JPO). The request specifies a JPO basic single payment in order to differentiate the transaction from a JPO one-time bonus payment. The default value of the **processingInformation. japanPaymentOptions. paymentMethod** field is `10`, which specifies a JPO single payment.
Endpoint:

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

Request

```
{
  "orderInformation": {
    "billTo": {
      "country": "US",
      "lastName": "Kim",
      "address1": "201 S. Division St.",
      "postalCode": "48104-2201",
      "locality": "Ann Arbor",
      "administrativeArea": "MI",
      "firstName": "Kyong-Jin",
      "email": "test@cybs.com"
    },
    "amountDetails": {
      "totalAmount": "100.00",
      "currency": "jpy"
    }
  },
  "paymentInformation": {
```

```
      "card": {
        "expirationYear": "2031",
        "number": "4111111111111111",
        "expirationMonth": "12",
        "type": "001"
      }
    },
    "processingInformation": {
      "japanPaymentOptions": {
        "businessName": "##",
        "businessNameAlphaNumeric": "OurStore",
        "businessNameKatakana": "######"
      }
    }
}
```

Response to a Successful Request

```
{
  "_links" : {
    "authReversal" : {
      "method" : "POST",
      "href" : "/pts/v2/payments/6842924689096191303059/reversals"
    },
    "self" : {
      "method" : "GET",
      "href" : "/pts/v2/payments/684292468909619130359"
    },
    "capture" : {
      "method" : "POST",
      "href" : "/pts/v2/payments/6842924689096191303059/captures"
    }
  },
  "clientReferenceInformation" : {
    "code" : "RTS-Auth"
  },
  "id" : "6842924689096191303059",
  "orderInformation" : {
    "invoiceDetails" : {
      "salesSlipNumber" : "52966"
    },
    "amountDetails" : {
      "authorizedAmount" : "100",
      "currency" : "jpy"
    }
  },
  "paymentAccountInformation" : {
    "card" : {
      "type" : "001"
    }
  },
  "paymentInformation" : {
    "tokenizedCard" : {
      "type" : "001"
    },
    "card" : {
```

```
    "type" : "001"
   }
  },
  "processorInformation" : {
   "salesSlipNumber" : "52966",
   "approvalCode" : "123456",
   "cardVerification" : {
    "resultCode" : "3"
   },
   "responseCategoryCode" : "000",
   "forwardedAcquirerCode" : "Sumitomo",
   "avs" : {
    "code" : "2"
   }
  },
  "reconciliationId" : "00202305171201090000000000001",
  "status" : "AUTHORIZED",
  "submitTimeUtc" : "2023-05-17T03:01:09Z"
}
```


Authorize a Bonus Payment with Japanese Payment Options

This section shows you how to process an authorization of a bonus payment with Japanese Payment Options (JPO).

Limitations

- The only supported acquirer is Sumitomo Mitsui Card Co.
- The payment must use a Visa payment card.

Prerequisites

- You have signed a contract with your acquirer.
- You have contacted your account provider for details about contracts and funding cycles. The funding cycle could differ when using JPO.
- Card holders who want to use JPO have signed a contract with an issuing bank.
- You have confirmed payment option availability with your account provider and card holder before implementing one of these payment options.

Endpoint

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

Required Fields for Authorizing a JPO Bonus Payment

Use these required fields for authorizing a JPO bonus payment.

> **Important**
>
> When using relaxed requirements for address data and the expiration date, not all fields in this list are required. It is your responsibility to determine whether your account is enabled to use this feature and which fields are required. For details about relaxed requirements, see *Relaxed Requirements for Address Data and Expiration Date in Payment Transactions* on page 264.

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.japanPaymentOptions.businessName

Business name in kanji characters.

processingInformation.japanPaymentOptions.businessNameAlphaNumeric

processingInformation.japanPaymentOptions.businessNameKatakana

processingInformation.japanPaymentOptions.paymentMethod

Set this field to 21, 22, 23, or 24.

processingInformation.japanPaymentOptions.terminalId

Required for card-present transactions. Unique Japan Credit Card Association (JCCA) terminal identifier that is provided by Cybersource.

Related Information

- *API field reference guide for the REST API*

REST Example: Authorizing a JPO Bonus Payment

This example shows an authorization request for a card-not-present bonus payment that uses Japanese payment options (JPO). The request specifies a JPO one-time bonus payment in order to differentiate the transaction from a JPO single payment.
Endpoint:

- Production: POST https://api.cybersource.com/pts/v2/payments
- Test: POST https://apitest.cybersource.com/pts/v2/payments

Request

```
{
  "orderInformation": {
    "billTo": {
      "country": "US",
      "lastName": "Kim",
      "address1": "201 S. Division St.",
      "postalCode": "48104-2201",
      "locality": "Ann Arbor",
      "administrativeArea": "MI",
      "firstName": "Kyong-Jin",
      "email": "test@cybs.com"
    },
    "amountDetails": {
      "totalAmount": "100.00",
      "currency": "jpy"
    }
  },
  "paymentInformation": {
    "card": {
      "expirationYear": "2031",
      "number": "4111111111111111",
      "expirationMonth": "12",
      "type": "001"
    }
  },
  "processingInformation": {
    "japanPaymentOptions": {
      "businessName": "##",
      "businessNameAlphaNumeric": "OurStore",
      "businessNameKatakana": "######",
      "paymentMethod": "21"
    }
  }
}
```

Response to a Successful Request

```
{
 "_links" : {
  "authReversal" : {
   "method" : "POST",
   "href" : "/pts/v2/payments/6843556498736135003059/reversals"
  },
```

```
    "self" : {
     "method" : "GET",
     "href" : "/pts/v2/payments/6843556498736135003059"
    },
    "capture" : {
     "method" : "POST",
     "href" : "/pts/v2/payments/6843556498736135003059/captures"
    }
   },
   "clientReferenceInformation" : {
    "code" : "RTS-Auth"
   },
   "id" : "6843556498736135003059",
   "orderInformation" : {
    "invoiceDetails" : {
     "salesSlipNumber" : "56307"
    },
    "amountDetails" : {
     "authorizedAmount" : "100",
     "currency" : "jpy"
    }
   },
   "paymentAccountInformation" : {
    "card" : {
     "type" : "001"
    }
   },
   "paymentInformation" : {
    "tokenizedCard" : {
     "type" : "001"
    },
    "card" : {
     "type" : "001"
    }
   },
   "processorInformation" : {
    "salesSlipNumber" : "56307",
    "approvalCode" : "123456",
    "cardVerification" : {
     "resultCode" : "3"
    },
    "responseCategoryCode" : "000",
    "forwardedAcquirerCode" : "Sumitomo",
    "avs" : {
     "code" : "2"
    }
   },
   "reconciliationId" : "00202305180534100000000000001",
   "status" : "AUTHORIZED",
   "submitTimeUtc" : "2023-05-17T20:34:10Z"
}
```

Authorize an Installment Payment with Japanese Payment Options

This section shows you how to process an authorization of an installment payment with Japanese Payment Options (JPO).

Limitations

- The only supported acquirer is Sumitomo Mitsui Card Co.
- The payment must use a Visa payment card.

Prerequisites

- You have signed a contract with your acquirer.
- You have contacted your account provider for details about contracts and funding cycles. The funding cycle could differ when using JPO.
- Card holders who want to use JPO have signed a contract with an issuing bank.
- You have confirmed payment option availability with your account provider and card holder before implementing one of these payment options.

Endpoint

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

Required Fields for Authorizing a JPO Installment Payment

Use these required fields for authorizing a JPO installment payment.

Important

When using relaxed requirements for address data and the expiration date, not all fields in this list are required. It is your responsibility to determine whether your account is enabled to use this feature and which fields are required. For details about relaxed requirements, see *Relaxed Requirements for Address Data and Expiration Date in Payment Transactions* on page 264.

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.japanPaymentOptions.businessName	**Business name in kanji characters.**
processingInformation.japanPaymentOptions.businessNameAlphaNumeric	
processingInformation.japanPaymentOptions.businessNameKatakana	
processingInformation.japanPaymentOptions.firstBillingMonth	**If you do not specify this field, it is set by default to the number of the next month.**
processingInformation.japanPaymentOptions.numberOfInstallments	**Number of monthly payments.**
processingInformation.japanPaymentOptions.paymentMethod	**Set the value to** `61`**.**
processingInformation.japanPaymentOptions.terminalId	**Required for card-present transactions. Unique Japan Credit Card Association (JCCA) terminal identifier that is provided by Cybersource.**

Related Information

- *API field reference guide for the REST API*

REST Example: Authorizing a JPO Installment Payment

This example shows an authorization request for a card-not-present installment payment that uses Japanese payment options (JPO). The first installment is paid in April, and 12 payments will be made.
Endpoint:

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

Request

```
{
  "orderInformation": {
    "billTo": {
      "country": "US",
      "lastName": "Kim",
      "address1": "201 S. Division St.",
```

```
        "postalCode": "48104-2201",
        "locality": "Ann Arbor",
        "administrativeArea": "MI",
        "firstName": "Kyong-Jin",
        "email": "test@cybs.com"
      },
      "amountDetails": {
        "totalAmount": "100.00",
        "currency": "jpy"
      }
    },
    "paymentInformation": {
      "card": {
        "expirationYear": "2031",
        "number": "4111111111111111",
        "expirationMonth": "12",
        "type": "001"
      }
    },
    "processingInformation": {
      "japanPaymentOptions": {
        "businessName": "##",
        "businessNameAlphaNumeric": "OurStore",
        "businessNameKatakana": "######",
        "firstBusinessMonth": "04",
        "installments": "12",
        "paymentMethod": "31"
      }
    }
}
```

Response to a Successful Request

```
{
  "_links" : {
    "authReversal" : {
      "method" : "POST",
      "href" : "/pts/v2/payments/6843585327946622203059/reversals"
    },
    "self" : {
      "method" : "GET",
      "href" : "/pts/v2/payments/684358532794662220 3059"
    },
    "capture" : {
      "method" : "POST",
      "href" : "/pts/v2/payments/6843585327946622203059/captures"
    }
  },
  "clientReferenceInformation" : {
    "code" : "RTS-Auth"
  },
  "id" : "6843585327946622203059",
  "orderInformation" : {
    "invoiceDetails" : {
      "salesSlipNumber" : "56311"
    },
```

```
    "amountDetails" : {
      "authorizedAmount" : "100",
      "currency" : "jpy"
    }
  },
  "paymentAccountInformation" : {
    "card" : {
      "type" : "001"
    }
  },
  "paymentInformation" : {
    "tokenizedCard" : {
      "type" : "001"
    },
    "card" : {
      "type" : "001"
    }
  },
  "processorInformation" : {
    "salesSlipNumber" : "56311",
    "approvalCode" : "123456",
    "cardVerification" : {
      "resultCode" : "3"
    },
    "responseCategoryCode" : "000",
    "forwardedAcquirerCode" : "Sumitomo",
    "avs" : {
      "code" : "2"
    }
  },
  "reconciliationId" : "00202305180622130000000000001",
  "status" : "AUTHORIZED",
  "submitTimeUtc" : "2023-05-17T21:22:13Z"
}
```


Authorize a Revolving Payment with Japanese Payment Options

This section shows you how to process an authorization of a revolving payment with Japanese Payment Options (JPO).

Limitations

- The only supported acquirer is Sumitomo Mitsui Card Co.
- The payment must use a Visa payment card.

Prerequisites

- You have signed a contract with your acquirer.

- You have contacted your account provider for details about contracts and funding cycles. The funding cycle could differ when using JPO.
- Card holders who want to use JPO have signed a contract with an issuing bank.
- You have confirmed payment option availability with your account provider and card holder before implementing one of these payment options.

Endpoint

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

Required Fields for Authorizing a Revolving Payment Using the JPO Method

Use these required fields for authorizing a revolving payment using the JPO method.

Important

When using relaxed requirements for address data and the expiration date, not all fields in this list are required. It is your responsibility to determine whether your account is enabled to use this feature and which fields are required. For details about relaxed requirements, see *Relaxed Requirements for Address Data and Expiration Date in Payment Transactions* on page 264.

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.japanPaymentOptions.businessName

Business name in kanji characters.

processingInformation.japanPaymentOptions.businessNameAlphaNumeric

processingInformation.japanPaymentOptions.businessNameKatakana	
processingInformation.japanPaymentOptions.firstBillingMonth	**Number of the month in which installment payments begin. The default value is the number of the month that follows the transaction date.**
processingInformation.japanPaymentOptions.installments	**Set this field to the number of installment payments.**
processingInformation.japanPaymentOptions.paymentMethod	**Set the value to `80`.**
processingInformation.japanPaymentOptions.terminalId	**Required for card-present transactions. Unique Japan Credit Card Association (JCCA) terminal identifier that is provided by Cybersource.**

Related Information

- *API field reference guide for the REST API*

REST Example: Authorizing a JPO Revolving Payment

This example shows an authorization request for a card-not-present revolving payment that uses Japanese payment options (JPO). The first installment is paid in May, and 12 payments will be made.
Endpoint:

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

Request

```
{
  "orderInformation": {
    "billTo": {
      "country": "US",
      "lastName": "Kim",
      "address1": "201 S. Division St.",
      "postalCode": "48104-2201",
      "locality": "Ann Arbor",
      "administrativeArea": "MI",
      "firstName": "Kyong-Jin",
      "email": "test@cybs.com"
    },
    "amountDetails": {
      "totalAmount": "100.00",
      "currency": "jpy"
    }
  },
  "paymentInformation": {
    "card": {
      "expirationYear": "2031",
      "number": "4111111111111111",
```

```
        "expirationMonth": "12",
        "type": "001"
      }
    },
    "processingInformation": {
      "japanPaymentOptions": {
        "businessName": "##",
        "businessNameAlphaNumeric": "OurStore",
        "businessNameKatakana": "######",
        "firstBusinessMonth": "05",
        "installments": "12",
        "paymentMethod": "80"
      }
    }
}
```

Response to a Successful Request

```
{
  "_links" : {
    "authReversal" : {
      "method" : "POST",
      "href" : "/pts/v2/payments/6843585327946622203059/reversals"
    },
    "self" : {
      "method" : "GET",
      "href" : "/pts/v2/payments/684358532794662220 3059"
    },
    "capture" : {
      "method" : "POST",
      "href" : "/pts/v2/payments/6843585327946622203059/captures"
    }
  },
  "clientReferenceInformation" : {
    "code" : "RTS-Auth"
  },
  "id" : "6843585327946622203059",
  "orderInformation" : {
    "invoiceDetails" : {
      "salesSlipNumber" : "56311"
    },
    "amountDetails" : {
      "authorizedAmount" : "100",
      "currency" : "jpy"
    }
  },
  "paymentAccountInformation" : {
    "card" : {
      "type" : "001"
    }
  },
  "paymentInformation" : {
    "tokenizedCard" : {
      "type" : "001"
    },
    "card" : {
```

```
    "type" : "001"
   }
  },
  "processorInformation" : {
   "salesSlipNumber" : "56311",
   "approvalCode" : "123456",
   "cardVerification" : {
    "resultCode" : "3"
   },
   "responseCategoryCode" : "000",
   "forwardedAcquirerCode" : "Sumitomo",
   "avs" : {
    "code" : "2"
   }
  },
  "reconciliationId" : "00202305180622130000000000001",
  "status" : "AUTHORIZED",
  "submitTimeUtc" : "2023-05-17T21:22:13Z"
}
```


Authorize a Combination Payment with Japanese Payment Options

This section shows you how to process an authorization of a combination bonus and installment payments with Japanese Payment Options (JPO).

Limitations

- The only supported acquirer is Sumitomo Mitsui Card Co.
- The payment must use a Visa payment card.

Prerequisites

- You have signed a contract with your acquirer.
- You have contacted your account provider for details about contracts and funding cycles. The funding cycle could differ when using JPO.
- Card holders who want to use JPO have signed a contract with an issuing bank.
- You have confirmed payment option availability with your account provider and card holder before implementing one of these payment options.

Endpoint

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

Required Fields for Authorizing a Combination Payment Using the JPO Method

Use these required fields for authorizing a combination payment using the JPO method.

> **Important**
>
> When using relaxed requirements for address data and the expiration date, not all fields in this list are required. It is your responsibility to determine whether your account is enabled to use this feature and which fields are required. For details about relaxed requirements, see *Relaxed Requirements for Address Data and Expiration Date in Payment Transactions* on page 264.

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.japanPaymentOptions.businessName
Business name in kanji characters.

processingInformation.japanPaymentOptions.businessNameAlphaNumeric

processingInformation.japanPaymentOptions.businessNameKatakana

processingInformation.japanPaymentOptions.firstBillingMonth

processingInformation.japanPaymentOptions.installments
Set this field to the number of monthly installments.

processingInformation.japanPaymentOptions.paymentMethod
Set this field to 31, 32, 33, **or** 34.

processingInformation.japanPaymentOptions.terminalId
Required for card-present transactions. Unique Japan Credit Card Association (JCCA) terminal identifier that is provided by Cybersource.

Related Information

- *API field reference guide for the REST API*

REST Example: Authorizing a JPO Combination Payment

This example shows an authorization request for a card-not-present Integrated bonus and installment payment that uses Japanese payment options (JPO). The first installment is paid in June, and 12 payments will be made.
Endpoint:

- Production: POST https://api.cybersource.com/pts/v2/payments
- Test: POST https://apitest.cybersource.com/pts/v2/payments

Request

```
{
  "orderInformation": {
    "billTo": {
      "country": "US",
      "lastName": "Kim",
      "address1": "201 S. Division St.",
      "postalCode": "48104-2201",
      "locality": "Ann Arbor",
      "administrativeArea": "MI",
      "firstName": "Kyong-Jin",
      "email": "test@cybs.com"
    },
    "amountDetails": {
      "totalAmount": "100.00",
      "currency": "jpy"
    }
  },
  "paymentInformation": {
    "card": {
      "expirationYear": "2031",
      "number": "4111111111111111",
      "expirationMonth": "12",
      "type": "001"
    }
  },
  "processingInformation": {
    "japanPaymentOptions": {
      "businessName": "##",
      "businessNameAlphaNumeric": "OurStore",
      "businessNameKatakana": "######",
      "firstBusinessMonth": "05",
      "installments": "12",
      "paymentMethod": "80"
    }
  }
}
```

Response to a Successful Request


```
{
 "_links" : {
  "authReversal" : {
   "method" : "POST",
   "href" : "/pts/v2/payments/6843585327946622203059/reversals"
  },
  "self" : {
   "method" : "GET",
   "href" : "/pts/v2/payments/684358532794662220 3059"
  },
  "capture" : {
   "method" : "POST",
   "href" : "/pts/v2/payments/6843585327946622203059/captures"
  }
 },
 "clientReferenceInformation" : {
  "code" : "RTS-Auth"
 },
 "id" : "6843585327946622203059",
 "orderInformation" : {
  "invoiceDetails" : {
   "salesSlipNumber" : "56311"
  },
  "amountDetails" : {
   "authorizedAmount" : "100",
   "currency" : "jpy"
  }
 },
 "paymentAccountInformation" : {
  "card" : {
   "type" : "001"
  }
 },
 "paymentInformation" : {
  "tokenizedCard" : {
   "type" : "001"
  },
  "card" : {
   "type" : "001"
  }
 },
 "processorInformation" : {
  "salesSlipNumber" : "56311",
  "approvalCode" : "123456",
  "cardVerification" : {
   "resultCode" : "3"
  },
  "responseCategoryCode" : "000",
  "forwardedAcquirerCode" : "Sumitomo",
  "avs" : {
   "code" : "2"
  }
 },
 "reconciliationId" : "00202305180622130000000000001",
```

```
  "status" : "AUTHORIZED",
  "submitTimeUtc" : "2023-05-17T21:22:13Z"
}
```

Level II Processing

This section shows you how to process transactions that include Level II data.

Related Information

- See *Level II and Level III Data* on page 41 for a description of and requirements for processing payments that include Level II data.

Additional Resources for Level II/III Payments

For more information, see these guides:

- *Level II and III Processing Developer Guide*
- *API field reference guide for the REST API*
- Github repositories: *https://github.com/Cybersource?q=rest-sample&type=all&language=&sort=*

Captures with Level II Data

This section shows you how to capture an authorized transaction with Level II data. These required fields and example are specific to Visa Platform Connect. For required fields, optional fields, and examples specific to your processor see the *Level II and Level III Processing developer guides*.

Endpoint

Production: `POST https://api.cybersource.com/pts/v2/payments/{id}/captures`
Test: `POST https://apitest.cybersource.com/pts/v2/payments/{id}/captures`
The `{id}` is the transaction ID returned in the authorization response.

Required Fields for Capturing a Payment with Level II Data

Use these required fields to capture a payment that includes Level II data.

orderInformation.amountDetails.currency	
orderInformation.amountDetails.nationalTaxIncluded	**Required if the sum of all orderInformation.lineItems[].taxDetails[].amount values = 0.**
orderInformation.amountDetails.totalAmount	
orderInformation.invoiceDetails.purchaseOrderNumber	**Required for purchase/procurement cards only.**
orderInformation.invoiceDetails.taxable	**Required if the sum of all orderInformation.lineItems.taxAmount values = 0.**

Related Information

- *API field reference guide for the REST API*

Optional Fields for Capturing a Payment with Level II Data

You can use these optional fields to include additional information when capturing a payment with Level II data.

order.vatTaxAmountSign

orderInformation.lineItems[].taxAmount

orderInformation.lineItems[].taxDetails[].amount

orderInformation.shipTo.postalCode

Related Information

- *API field reference guide for the REST API*

REST Example: Capturing a Payment with Level II Data

Endpoint:

- Production: `POST https://api.cybersource.com/pts/v2/payments/{id}/captures`
- Test: `POST https://apitest.cybersource.com/pts/v2/payments/{id}/captures`

The `{id}` is the transaction ID returned in the authorization response.

Request

```
{
  "clientReferenceInformation": {
    "code": "TC50171_3"
```

```
  },
  "orderInformation": {
    "amountDetails": {
      "totalAmount": "102.21",
      "currency": "USD"
    }
  }
}
```

Response to a Successful Request

```
{
  "_links": {
    "void": {
      "method": "POST",
      "href": "/pts/v2/captures/6807039415896954303954/voids"
    },
    "self": {
      "method": "GET",
      "href": "/pts/v2/captures/6807039415896954303954"
    }
  },
  "clientReferenceInformation": {
    "code": "TC50171_3"
  },
  "id": "6807039415896954303954",
  "orderInformation": {
    "amountDetails": {
      "totalAmount": "102.21",
      "currency": "USD"
    }
  },
  "reconciliationId": "680703588213683080 3954",
  "status": "PENDING",
  "submitTimeUtc": "2023-04-05T14:12:21Z"
}
```

Credits with Level II Data

This topic shows you how to process a credit with Level II data. These required fields and example are specific to Visa Platform Connect. For required fields, optional fields, and examples specific to your processor see the *Level II and Level III Processing developer guides*.

Endpoint

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

Required Fields for Processing a Credit with Level II Data

Use these required fields to process a credit that includes Level II data.

> **Important**
>
> When using relaxed requirements for address data and the expiration date, not all fields in this list are required. It is your responsibility to determine whether your account is enabled to use this feature and which fields are required. For details about relaxed requirements, see *Relaxed Requirements for Address Data and Expiration Date in Payment Transactions* on page 264.

orderInformation.amountDetails.currency	
orderInformation.amountDetails.nationalTaxIncluded	**Required if the sum of all orderInformation.lineItems[].taxDetails[].amount values = 0.**
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	
orderInformation.invoiceDetails.purchaseOrderNumber	**Required for purchase/procurement cards only.**
orderInformation.invoiceDetails.taxable	**Required if the sum of all orderInformation.lineItems.taxAmount values = 0.**
paymentInformation.card.expirationMonth	
paymentInformation.card.expirationYear	
paymentInformation.card.number	

Related Information

- *API field reference guide for the REST API*

Optional Fields for Processing a Credit with Level II Data

You can use these optional fields to include additional information when processing a credit request with Level II data.

order.vatTaxAmountSign

orderInformation.lineItems[].taxAmount

orderInformation.lineItems[].taxDetails[].amount

orderInformation.shipTo.postalCode

Related Information

- *API field reference guide for the REST API*

REST Example: Processing a Credit with Level II Data

Endpoint:

- Production: POST https://api.cybersource.com/pts/v2/credits/
- Test: POST https://apitest.cybersource.com/pts/v2/credits/

Request

```
{
 "paymentInformation": {
  "card": {
   "number": "41111111XXXXXXXX",
   "expirationMonth": "03",
   "expirationYear": "2031"
  }
 },
 "orderInformation": {
  "amountDetails": {
   "totalAmount": "200",
   "currency": "USD"
  },
  "billTo": {
   "firstName": "John",
   "lastName": "Deo",
   "address1": "900 Metro Center Blvd",
   "locality": "Foster City",
   "administrativeArea": "CA",
   "postalCode": "48104-2201",
   "country": "US",
   "email": "test@cybs.com",
   "phoneNumber": "9321499232"
  }
 }
}
```

Response to a Successful Request


```
{
  "_links": {
    "void": {
      "method": "POST",
      "href": "/pts/v2/credits/6807043651016112003955/voids"
    },
    "self": {
      "method": "GET",
      "href": "/pts/v2/credits/6807043651016112003955"
    }
  },
  "clientReferenceInformation": {
    "code": "1680704365197"
  },
  "creditAmountDetails": {
    "currency": "USD",
    "creditAmount": "200.00"
  },
  "id": "6807043651016112003955",
  "orderInformation": {
    "amountDetails": {
      "currency": "USD"
    }
  },
  "paymentAccountInformation": {
    "card": {
      "type": "001"
    }
  },
  "paymentInformation": {
    "tokenizedCard": {
      "type": "001"
    },
    "card": {
      "type": "001"
    }
  },
  "processorInformation": {
    "approvalCode": "831000",
    "networkTransactionId": "016153570198200",
    "retrievalReferenceNumber": "309514100806",
    "responseCode": "00"
  },
  "reconciliationId": "6807043651016112003955",
  "status": "PENDING",
  "submitTimeUtc": "2023-04-05T14:19:25Z"
}
```

Sales with Level II Data

This section shows you how to process a sale transaction with Level II data. These required fields and example are specific to Visa Platform Connect. For required fields, optional fields, and examples specific to your processor see the *Level II and Level III Processing developer guides*.
A sale transaction combines and authorization and a capture into a single transaction.

Endpoint

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

Required Fields for Processing a Sale with Level II Data

Use these required fields to process a sale that includes Level II data.

Important

When using relaxed requirements for address data and the expiration date, not all fields in this list are required. It is your responsibility to determine whether your account is enabled to use this feature and which fields are required. For details about relaxed requirements, see *Relaxed Requirements for Address Data and Expiration Date in Payment Transactions* on page 264.

orderInformation.amountDetails.currency	
orderInformation.amountDetails.nationalTaxIncluded	**Required if the sum of all orderInformation.lineItems[].taxDetails[].amount values = 0.**
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	
orderInformation.invoiceDetails.purchaseOrderNumber	**Required for purchase/procurement cards only.**

orderInformation.invoiceDetails.taxable	**Required if the sum of all orderInformation.lineItems.taxAmount values = `0`.**
paymentInformation.card.expirationMonth	
paymentInformation.card.expirationYear	
paymentInformation.card.number	
paymentInformation.card.securityCode	**Required only for Visa Platform Connect.**
paymentInformation.card.type	
processingInformation.capture	**Set field to `true`.**

Related Information

- *API field reference guide for the REST API*

Optional Fields for Processing a Sale with Level II Data

You can use these optional fields to include additional information when processing a sale with Level II data.

order.vatTaxAmountSign

orderInformation.lineItems[].taxAmount

orderInformation.lineItems[].taxDetails[].amount

orderInformation.shipTo.postalCode

Related Information

- *API field reference guide for the REST API*

REST Example: Processing a Sale with Level II Data

Endpoint:

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

Request

```
{
  "processingInformation": {
    "capture": true
  },
  "orderInformation" : {
    "billTo" : {
    "country" : "US",
    "lastName" : "VDP",
    "address1" : "201 S. Division St.",
```

```
    "postalCode" : "48104-2201",
    "locality" : "Ann Arbor",
    "administrativeArea" : "MI",
    "firstName" : "RTS",
    "email" : "test@cybs.com"
   },
    "amountDetails" : {
     "totalAmount" : "100.00",
     "currency" : "usd"
    }
   },
  "paymentInformation" : {
    "card" : {
     "expirationYear" : "2031",
     "number" : "4111111111111111",
     "expirationMonth" : "12",
     "type" : "001",
     "securityCode": "999"


   }
  }
}
```

Response to a Successful Request

```
{
  "_links": {
    "void": {
      "method": "POST",
      "href": "/pts/v2/payments/680704701070687270 3954/voids"
    },
    "self": {
      "method": "GET",
      "href": "/pts/v2/payments/6807047010706872703954"
    }
  },
  "clientReferenceInformation": {
    "code": "1680704701200"
  },
  "id": "6807047010706872703954",
  "orderInformation": {
    "amountDetails": {
      "totalAmount": "100.00",
      "authorizedAmount": "100.00",
      "currency": "usd"
    }
  },
  "paymentAccountInformation": {
    "card": {
      "type": "001"
    }
  },
  "paymentInformation": {
    "tokenizedCard": {
      "type": "001"
```

```
    },
    "card": {
      "type": "001"
    }
  },
  "processorInformation": {
    "systemTraceAuditNumber": "101247",
    "approvalCode": "831000",
    "cardVerification": {
      "resultCodeRaw": "M",
      "resultCode": "M"
    },
    "merchantAdvice": {
      "code": "01",
      "codeRaw": "M001"
    },
    "responseDetails": "ABC",
    "networkTransactionId": "016153570198200",
    "retrievalReferenceNumber": "309514101247",
    "consumerAuthenticationResponse": {
      "code": "2",
      "codeRaw": "2"
    },
    "transactionId": "016153570198200",
    "responseCode": "00",
    "avs": {
      "code": "Y",
      "codeRaw": "Y"
    }
  },
  "reconciliationId": "6807047010706872703954",
  "status": "AUTHORIZED",
  "submitTimeUtc": "2023-04-05T14:25:01Z"
}
```

Level III Processing

This section shows you how to process transactions that include Level III data.

- See *Level II and Level III Data* on page 41 for a description of and requirements for processing payments that include Level III data.

Additional Resources for Level II/III Payments

For more information, see these guides:

- *Level II and III Processing Developer Guide*
- *API field reference guide for the REST API*
- Github repositories: *https://github.com/Cybersource?q=rest-sample&type=all&language=&sort=*

Captures with Level III Data

This section shows you how to capture an authorized transaction with Level III data. These required fields and example are specific to Visa Platform Connect. For required fields, optional fields, and examples specific to your processor see the *Level II and Level III Processing developer guides*.

Endpoint

Production: `POST https://api.cybersource.com/pts/v2/payments/{id}/captures`
Test: `POST https://apitest.cybersource.com/pts/v2/payments/{id}/captures`
The `{id}` is the transaction ID returned in the authorization response.

Required Fields for Capturing a Payment with Level III Data

Use these required fields to capture a payment that includes Level III data.

orderInformation.amountDetails.currency	
orderInformation.amountDetails.nationalTaxIncluded	**Required if the sum of all orderInformation.lineItems[].taxDetails[].amount values = 0.**
orderInformation.amountDetails.totalAmount	
orderInformation.invoiceDetails.purchaseOrderNumber	**Required for purchase/procurement cards only.**
orderInformation.invoiceDetails.taxable	**Required if the sum of all orderInformation.lineItems.taxAmount values = 0.**
processingInformation.purchaseLevel	**Set field to 3.**

Related Information

- *API field reference guide for the REST API*

Optional Fields for Capturing a Payment with Level III Data

You can use these optional fields to include additional information when capturing a payment with Level III data.

buyerInformation.vatRegistrationNumber

merchantInformation.cardAcceptorReferenceNumber

merchantInformation.vatRegistrationNumber

order.vatTaxAmountSign

orderInformation.amountDetails.discountAmount

orderInformation.amountDetails.dutyAmount

orderInformation.amountDetails.freightAmount

orderInformation.amountDetails.taxAppliedAfterDiscount

orderInformation.amountDetails.taxAppliedLevel

orderInformation.amountDetails.taxDetails[].amount

orderInformation.amountDetails.taxDetails[].rate

orderInformation.invoiceDetails.commodityCode

orderInformation.invoiceDetails.purchaseContactName

orderInformation.invoiceDetails.purchaseOrderDate

orderInformation.invoiceDetails.vatInvoiceReferenceNumber

orderInformation.lineItems[].commodityCode

orderInformation.lineItems[].discountAmount

orderInformation.lineItems[].discountRate

orderInformation.lineItems[].invoiceNumber

orderInformation.lineItems[].productCode

orderInformation.lineItems[].productName

orderInformation.lineItems[].quantity

orderInformation.lineItems[].taxAmount

orderInformation.lineItems[].taxAppliedAfterDiscount

orderInformation.lineItems[].taxDetails[].amount

orderInformation.lineItems[].taxRate

orderInformation.lineItems[].taxStatusIndicator

orderInformation.lineItems[].totalAmount

orderInformation.lineItems[].typeOfSupply

orderInformation.lineItems[].unitOfMeasure

orderInformation.lineItems[].unitPrice

orderInformation.shippingDetails.shipFromPostalCode

orderInformation.shipTo.administrativeArea

orderInformation.shipTo.postalCode

senderInformation.vatRegistrationNumber

Related Information

- *API field reference guide for the REST API*

REST Example: Capturing a Payment with Level III Data

Endpoint:

- Production: `POST https://api.cybersource.com/pts/v2/payments/{id}/captures`
- Test: `POST https://apitest.cybersource.com/pts/v2/payments/{id}`

The `{id}` is the transaction ID returned in the authorization response.

Request

```
{
  "processingInformation": {
    "purchaseLevel": "3"
  },
  "orderInformation": {
    "amountDetails": {
      "totalAmount": "100.00",
      "currency": "USD"
    }
```

```
  }
}
```

Response to a Successful Request

```
{
  "_links": {
    "void": {
      "method": "POST",
      "href": "/pts/v2/captures/6807093180006195204953/voids"
    },
    "self": {
      "method": "GET",
      "href": "/pts/v2/captures/6807093180006195204953"
    }
  },
  "clientReferenceInformation": {
    "code": "1680709318092"
  },
  "id": "6807093180006195204953",
  "orderInformation": {
    "amountDetails": {
      "totalAmount": "100.00",
      "currency": "USD"
    }
  },
  "reconciliationId": "6807091257096356004951",
  "status": "PENDING",
  "submitTimeUtc": "2023-04-05T15:41:58Z"
}
```

Credits with Level III Data

This topic shows you how to process a credit with Level III data. These required fields and example are specific to Visa Platform Connect. For required fields, optional fields, and examples specific to your processor see the *Level II and Level III Processing developer guides*.

Endpoint

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

Required Fields for Processing a Credit with Level III Data

Use these required fields to process a credit that includes Level III data.

orderInformation.amountDetails.currency

orderInformation.amountDetails.nationalTaxIncluded	**Required if the sum of all orderInformation.lineItems[].taxDetails[].amount values = 0.**
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	
orderInformation.invoiceDetails.purchaseOrderNumber	**Required for purchase/procurement cards only.**
orderInformation.invoiceDetails.taxable	**Required if the sum of all orderInformation.lineItems.taxAmount values = 0.**
paymentInformation.card.expirationMonth	
paymentInformation.card.expirationYear	
paymentInformation.card.number	
processingInformation.purchaseLevel	**Set field to 3.**

Related Information

- *API field reference guide for the REST API*

Optional Fields for Processing a Credit with Level III Data

You can use these optional fields to include additional information when processing a credit with Level III data.

buyerInformation.vatRegistrationNumber

merchantInformation.cardAcceptorReferenceNumber

merchantInformation.vatRegistrationNumber

order.vatTaxAmountSign

orderInformation.amountDetails.discountAmount

orderInformation.amountDetails.dutyAmount

orderInformation.amountDetails.freightAmount

orderInformation.amountDetails.taxAppliedAfterDiscount

orderInformation.amountDetails.taxAppliedLevel

orderInformation.amountDetails.taxDetails[].amount

orderInformation.amountDetails.taxDetails[].rate

orderInformation.invoiceDetails.commodityCode

orderInformation.invoiceDetails.purchaseContactName

orderInformation.invoiceDetails.purchaseOrderDate

orderInformation.invoiceDetails.vatInvoiceReferenceNumber

orderInformation.lineItems[].commodityCode

orderInformation.lineItems[].discountAmount

orderInformation.lineItems[].discountRate

orderInformation.lineItems[].invoiceNumber

orderInformation.lineItems[].productCode

orderInformation.lineItems[].productName

orderInformation.lineItems[].quantity

orderInformation.lineItems[].taxAmount

orderInformation.lineItems[].taxAppliedAfterDiscount

orderInformation.lineItems[].taxDetails[].amount

orderInformation.lineItems[].taxRate

orderInformation.lineItems[].taxStatusIndicator

orderInformation.lineItems[].totalAmount

orderInformation.lineItems[].typeOfSupply

orderInformation.lineItems[].unitOfMeasure

orderInformation.lineItems[].unitPrice

orderInformation.shippingDetails.shipFromPostalCode

orderInformation.shipTo.administrativeArea

orderInformation.shipTo.postalCode

senderInformation.vatRegistrationNumber

Related Information

- *API field reference guide for the REST API*

REST Example: Processing a Credit with Level III Data

Endpoint:

- Production: POST https://api.cybersource.com/pts/v2/credits/
- Test: POST https://apitest.cybersource.com/pts/v2/credits/

Request

```
{
 "processingInformation": {
  "purchaseLevel": "3"
 },
 "paymentInformation": {
  "card": {
   "number": "411111111111XXXX",
   "expirationMonth": "12",
   "expirationYear": "2031"
  }
 },
 "orderInformation": {
  "amountDetails": {
   "totalAmount": "100.00",
   "currency": "usd"
  },
  "billTo": {
   "firstName": "RTS",
   "lastName": "VDP",
   "address1": "201 S. Division St.",
   "locality": "Ann Arbor",
   "administrativeArea": "MI",
   "postalCode": "48104-2201",
   "country": "US",
   "email": "test@cybs.com"
  }
 }
}
```

Response to a Successful Request

```
{
 "_links": {
  "void": {
   "method": "POST",
   "href": "/pts/v2/credits/6807067080966508003954/voids"
  },
  "self": {
   "method": "GET",
   "href": "/pts/v2/credits/6807067080966508003954"
  }
 },
 "clientReferenceInformation": {
  "code": "1680706708181"
 },
 "creditAmountDetails": {
  "currency": "usd",
```

```
    "creditAmount": "100.00"
  },
  "id": "6807067080966508003954",
  "orderInformation": {
   "amountDetails": {
    "currency": "usd"
   }
  },
  "paymentAccountInformation": {
   "card": {
    "type": "001"
   }
  },
  "paymentInformation": {
   "tokenizedCard": {
    "type": "001"
   },
   "card": {
    "type": "001"
   }
  },
  "processorInformation": {
   "approvalCode": "831000",
   "networkTransactionId": "016153570198200",
   "retrievalReferenceNumber": "309514102853",
   "responseCode": "00"
  },
  "reconciliationId": "6807067080966508003954",
  "status": "PENDING",
  "submitTimeUtc": "2023-04-05T14:58:28Z"
}
```


Sales with Level III Data

This section shows you how to process a sale transaction with Level III data. These required fields and example are specific to Visa Platform Connect. For required fields, optional fields, and examples specific to your processor see the *Level II and Level III Processing developer guides*.
A sale transaction combines and authorization and a capture into a single transaction.

Endpoint

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

Required Fields for Processing a Sale with Level III Data

Use these required fields to process a sale that includes Level III data.

Important

> When using relaxed requirements for address data and the expiration date, not all fields in this list are required. It is your responsibility to determine whether your account is enabled to use this feature and which fields are required. For details about relaxed requirements, see *Relaxed Requirements for Address Data and Expiration Date in Payment Transactions* on page 264.

orderInformation.amountDetails.currency	
orderInformation.amountDetails.nationalTaxIncluded	**Required if the sum of all orderInformation.lineItems[].taxDetails[].amount values = `0`.**
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	
orderInformation.invoiceDetails.purchaseOrderNumber	**Required for purchase/procurement cards only.**
orderInformation.invoiceDetails.taxable	**Required if the sum of all orderInformation.lineItems.taxAmount values = `0`.**
paymentInformation.card.expirationMonth	
paymentInformation.card.expirationYear	
paymentInformation.card.number	
paymentInformation.card.securityCode	**Required only for Visa Platform Connect.**
paymentInformation.card.type	
processingInformation.capture	**Set field to `true`.**
processingInformation.purchaseLevel	**Set field to `3`.**

Related Information

- *API field reference guide for the REST API*

Optional Fields for Processing a Sale with Level III Data

You can use these optional fields to include additional information when processing a sale request with Level III data.

buyerInformation.vatRegistrationNumber

merchantInformation.cardAcceptorReferenceNumber

merchantInformation.vatRegistrationNumber

order.vatTaxAmountSign

orderInformation.amountDetails.discountAmount

orderInformation.amountDetails.dutyAmount

orderInformation.amountDetails.freightAmount

orderInformation.amountDetails.taxAppliedAfterDiscount

orderInformation.amountDetails.taxAppliedLevel

orderInformation.amountDetails.taxDetails[].amount

orderInformation.amountDetails.taxDetails[].rate

orderInformation.invoiceDetails.commodityCode

orderInformation.invoiceDetails.purchaseContactName

orderInformation.invoiceDetails.purchaseOrderDate

orderInformation.invoiceDetails.vatInvoiceReferenceNumber

orderInformation.lineItems[].commodityCode

orderInformation.lineItems[].discountAmount

orderInformation.lineItems[].discountRate

orderInformation.lineItems[].invoiceNumber

orderInformation.lineItems[].productCode

orderInformation.lineItems[].productName

orderInformation.lineItems[].quantity

orderInformation.lineItems[].taxAmount

orderInformation.lineItems[].taxAppliedAfterDiscount

orderInformation.lineItems[].taxDetails[].amount

orderInformation.lineItems[].taxRate

orderInformation.lineItems[].taxStatusIndicator

orderInformation.lineItems[].totalAmount

orderInformation.lineItems[].typeOfSupply

orderInformation.lineItems[].unitOfMeasure

orderInformation.lineItems[].unitPrice

orderInformation.shippingDetails.shipFromPostalCode

orderInformation.shipTo.administrativeArea

orderInformation.shipTo.postalCode

senderInformation.vatRegistrationNumber

Related Information

- *API field reference guide for the REST API*

REST Example: Processing a Sale with Level III Data

Endpoint:

- Production: POST https://api.cybersource.com/pts/v2/payments
- Test: POST https://apitest.cybersource.com/pts/v2/payments

Request

```
{
 "clientReferenceInformation": {
  "code": "TC50171_3"
 },
 "processingInformation": {
  "capture": true,
  "purchaseLevel": "3"
 },
 "paymentInformation": {
  "card": {
   "number": "41111111XXXXXXXX",
   "expirationMonth": "12",
   "expirationYear": "2031",
   "securityCode": "999"
  }
 },
 "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"
  }
 }
```

```
}
```

Response to a Successful Request

```
{
 "_links": {
  "void": {
   "method": "POST",
   "href": "/pts/v2/payments/6847868492196261203954/voids"
  },
  "self": {
   "method": "GET",
   "href": "/pts/v2/payments/6847868492196261203954"
  }
 },
 "clientReferenceInformation": {
  "code": "TC50171_3"
 },
 "id": "6847868492196261203954",
 "orderInformation": {
  "amountDetails": {
   "totalAmount": "102.21",
   "authorizedAmount": "102.21",
   "currency": "USD"
  }
 },
 "paymentAccountInformation": {
  "card": {
   "type": "001"
  }
 },
 "paymentInformation": {
  "tokenizedCard": {
   "type": "001"
  },
  "card": {
   "type": "001"
  }
 },
 "processorInformation": {
  "systemTraceAuditNumber": "152662",
  "approvalCode": "831000",
  "cardVerification": {
   "resultCodeRaw": "M",
   "resultCode": "M"
  },
  "merchantAdvice": {
   "code": "01",
   "codeRaw": "M001"
  },
  "responseDetails": "ABC",
  "networkTransactionId": "016153570198200",
  "retrievalReferenceNumber": "314220152662",
  "consumerAuthenticationResponse": {
   "code": "2",
   "codeRaw": "2"
```

```
    },
    "transactionId": "016153570198200",
    "responseCode": "00",
    "avs": {
      "code": "Y",
      "codeRaw": "Y"
    }
  },
  "reconciliationId": "6847868492196261203954",
  "status": "AUTHORIZED",
  "submitTimeUtc": "2023-05-22T20:20:49Z"
}
```

Mastercard Processing

These use cases are specific to Mastercard processing.

Mastercard Bill Payment Processing

This section describes how to request an authorization for a Mastercard Bill Payment.

Field Specific to this Use Case

Include this field with a standard authorization request when processing a Mastercard Bill Payment:

processingInformation.authorizationOptions.billPaymentType	**Set the value to indicate the type of bill that the cardholder is paying.**

Requirements

Sign up with Mastercard to participate in their bill payment program.

Endpoint

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

Related Information

- See *Mastercard Bill Payments* on page 42 for a description of and requirements for processing Mastercard Bill Payments.

Required Fields for Authorizing a Mastercard Bill Payment

Use these required fields to authorize a Mastercard bill payment.

Important

When using relaxed requirements for address data and the expiration date, not all fields in this list are required. It is your responsibility to determine whether your account is enabled to use this feature and which fields are required. For details about relaxed requirements, see *Relaxed Requirements for Address Data and Expiration Date in Payment Transactions* on page 264.

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

processingInformation.authorizationOptions.billPayment.type **Set the value to indicate the type of bill that the cardholder is paying.**

Related Information

- *API field reference guide for the REST API*

REST Example: Authorizing a Mastercard Bill Payment

This example shows a successful authorization request for a Mastercard bill payment.

Endpoint

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

Request

In this example, the cardholder in Brazil is using a Mastercard payment card (card type code 002) to pay a utility bill (Mastercard bill payment type value 001).

```
{
  "orderInformation": {
    "billTo": {
      "country": "BR",
      "lastName": "Doe",
```

```
        "firstName": "John",
        "address1": "Av Pres Juscelino Kubistchek 1909",
        "address2": "",
        "postalCode": "04543907",
        "locality": "Sao Paulo",
        "administrativeArea": "SP",
        "email": "john.doe@company.com"
      },
      "amountDetails": {
        "totalAmount": "100.00",
        "currency": "BRL"
      }
    },
    "paymentInformation": {
      "card": {
        "expirationMonth": "12",
        "expirationYear": "2031",
        "number": "555555555555xxxx",
        "securityCode": "123",
        "type": "002"
      }
    },
    "processingInformation": {
      "authorizationOptions": {
        "billPaymentType": "001"
      }
    }
}
```

Response to a Successful Request

```
{
  "_links" : {
    "authReversal" : {
      "method" : "POST",
      "href" : "/pts/v2/payments/6863356803746501803955/reversals"
    },
    "self" : {
      "method" : "GET",
      "href" : "/pts/v2/payments/686335680374650180 3955"
    },
    "capture" : {
      "method" : "POST",
      "href" : "/pts/v2/payments/6863356803746501803955/captures"
    }
  },
  "clientReferenceInformation" : {
    "code" : "1686335680358"
  },
  "id" : "6863356803746501803955",
  "orderInformation" : {
    "amountDetails" : {
      "authorizedAmount" : "100.00",
      "currency" : "brl"
    }
  },
```

```
  "paymentAccountInformation" : {
   "card" : {
    "type" : "002"
   }
  },
  "paymentInformation" : {
   "tokenizedCard" : {
    "type" : "002"
   },
   "card" : {
    "type" : "002"
   }
  },
  "processorInformation" : {
   "approvalCode" : "010012",
   "networkTransactionId" : "999010012",
   "transactionId" : "72b2900a9f316142b627a21031b48b0c259f08ffba0004172a04450c5d212345",
   "responseCode" : "400",
   "avs" : {
    "code" : "2"
   }
  },
  "reconciliationId" : "NHRRGOVtUxkb",
  "status" : "AUTHORIZED",
  "submitTimeUtc" : "2023-06-09T18:34:40Z"
}
```


Response to a Successful Request


```
{
  "_links" : {
   "authReversal" : {
    "method" : "POST",
    "href" : "/pts/v2/payments/6863354737666316003955/reversals"
   },
   "self" : {
    "method" : "GET",
    "href" : "/pts/v2/payments/686335473766631600 3955"
   },
   "capture" : {
    "method" : "POST",
    "href" : "/pts/v2/payments/6863354737666316003955/captures"
   }
  },
  "clientReferenceInformation" : {
   "code" : "1686335473749",
   "partner" : {
    "solutionId" : "89012345"
   }
  },
  "id" : "6863354737666316003955",
  "orderInformation" : {
   "amountDetails" : {
    "authorizedAmount" : "100.00",
    "currency" : "brl"
   }
```

```
  },
  "paymentAccountInformation" : {
    "card" : {
      "type" : "002"
    }
  },
  "paymentInformation" : {
    "tokenizedCard" : {
      "type" : "002"
    },
    "card" : {
      "type" : "002"
    }
  },
  "processorInformation" : {
    "approvalCode" : "010012",
    "networkTransactionId" : "999010012",
    "transactionId" : "72b2900a9f316142b627a21031b48b0c259f08ffba0004172a04450c5d212345",
    "responseCode" : "400",
    "avs" : {
      "code" : "2"
    }
  },
  "reconciliationId" : "q1bqrBcLeuTB",
  "status" : "AUTHORIZED",
  "submitTimeUtc" : "2023-06-09T18:31:13Z"
}
```


Mastercard Expert Monitoring Solutions Processing

This section shows you how to obtain the transaction fraud score assigned by Mastercard Expert Monitoring Solutions.

Requirement

Contact customer support to enable Mastercard Expert Monitoring Solutions for your account.

Important

After this feature is enabled for your account, Mastercard returns a fraud score for all your card-not-present authorization requests for Mastercard payment cards issued in the US.

Endpoint

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

Related Information

- See *Mastercard Expert Monitoring Solutions* on page 42 for a description of the transaction fraud score determined by Mastercard Expert Monitoring Solutions.

Required Fields for Processing an Authorization with Mastercard Expert Monitoring Solutions

Use these required fields to process an authorization using Mastercard Expert Monitoring Solutions.

> **Important**
>
> When using relaxed requirements for address data and the expiration date, not all fields in this list are required. It is your responsibility to determine whether your account is enabled to use this feature and which fields are required. For details about relaxed requirements, see *Relaxed Requirements for Address Data and Expiration Date in Payment Transactions* on page 264.

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

Related Information

- *API field reference guide for the REST API*

Response Field for Authorizations with Mastercard Expert Monitoring Solutions

This field can be returned in a response to an authorization using Mastercard Expert Monitoring Solutions.

processorInformation.emsTransactionRiskScore	**Fraud score for a Mastercard transaction.**

Related Information

- *API field reference guide for the REST API*

REST Example: Obtaining the Mastercard Fraud Score for an Authorization

Endpoint:

- Production: POST https://api.cybersource.com/pts/v2/payments
- Test: POST https://apitest.cybersource.com/pts/v2/payments

This example shows a successful authorization request for a Mastercard payment card issued in the US. If Mastercard Expert Monitoring Solutions is enabled, Mastercard can return a fraud score for the transaction.

Request

```
{
  "orderInformation": {
    "billTo": {
      "country": "US",
      "lastName": "Kim",
      "address1": "201 S. Division St.",
      "postalCode": "48104-2201",
      "locality": "Ann Arbor",
      "administrativeArea": "MI",
      "firstName": "Kyong-Jin",
      "email": "kim.test@cybs.com"/>"
    },
    "amountDetails": {
      "totalAmount": "100.00",
      "currency": "usd"
    }
  },
  "paymentInformation": {
    "card": {
      "number": "555555555555xxxx",
      "expirationYear": "2031",
      "expirationMonth": "12",
      "type": "002"
    }
  }
}
```

Response to a Successful Request

The **processorInformation.emsTransactionRiskScore** response field contains the fraud score returned by Mastercard Expert Monitoring Solutions. In this example, the fraud

score indicates a high likelihood (field value 843) of suspicious service station activity (field value 09).


```
{
 "_links" : {
  "authReversal" : {
   "method" : "POST",
   "href" : "/pts/v2/payments/6461731521426399003473/reversals"
  },
  "self" : {
   "method" : "GET",
   "href" : "/pts/v2/payments/646173152142639903473"
  },
  "capture" : {
   "method" : "POST",
   "href" : "/pts/v2/payments/6461731521426399003473/captures"
  }
 },
 "clientReferenceInformation" : {
  "code" : "1646173152047"
 },
 "id" : "646173152142639903473",
 "orderInformation" : {
  "amountDetails" : {
   "authorizedAmount" : "100.00",
   "currency" : "usd"
  }
 },
 "paymentAccountInformation" : {
  "card" : {
   "type" : "002"
  }
 },
 "paymentInformation" : {
  "tokenizedCard" : {
   "type" : "002"
  },
  "card" : {
   "type" : "002"
  }
 },
"paymentInsightsInformation" : {
  "responseInsights" : {
   "categoryCode" : "01"
  }
 },
 "processorInformation" : {
  "emsTransactionRiskScore": "84309",
  "systemTraceAuditNumber" : "862481",
  "approvalCode" : "831000",
  "merchantAdvice" : {
   "code" : "01",
   "codeRaw" : "M001"
  },
  "responseDetails" : "ABC",
  "networkTransactionId" : "016153570198200",
```

```
    "consumerAuthenticationResponse" : {
      "code" : "2",
      "codeRaw" : "2"
    },
    "transactionId" : "016153570198200",
    "responseCode" : "00",
    "avs" : {
      "code" : "Y",
      "codeRaw" : "Y"
    }
  },
  "reconciliationId" : "6461731521426399003473",
  "status" : "AUTHORIZED",
  "submitTimeUtc" : "2023-06-09T22:19:12Z"
}
```

Payer Authentication Processing

This section shows you how to process authorizations that use these payer authentication methods:

- American Express: SafeKey
- JCB: J/Secure
- Mastercard: Identity Check
- Visa: Visa Secure

Related Information

- See the*Payer Authentication Developer Guide* for details about payer authentication.

Additional Resources for Payer Authentication

For more information, see these guides:

- *Payer Authentication Developer Guide*
- *API field reference guide for the REST API*
- Github repositories: *https://github.com/Cybersource?q=rest-sample&type=all&language=&sort=*

Providing Payer Authentication Information for Authorization

The values that are returned from payer authentication must be provided when seeking authorization for the transaction. Authentication information that is not included when considering authorization may cause the transaction to be refused or downgraded and prevent the normal liability shift from occurring.
The level of security in payer authentication is denoted by the two digit Electronic Commerce Indicator (ECI) that is assigned to the transaction. These digital values have text equivalents which are assigned to the **processingInformation.commerceIndicator** field.

The America Express, Diners, Discover, UPI, and Visa card brands use 05, 06, and 07 digit values to express the authentication level for a 3#D Secure transaction.

Text Values for ECI Values

ECI Value	Meaning	Visa	Diners	Discover	UPI	Amex
05	Authenticated	vbv	pb	dipb	up3ds	aesk
06	Attempted authentication with a cryptogram	vbv_attempted	pb_attempted	dipb_attempted	up3ds_attempted	aesk_attempted
07	Internet, not authenticated	vbv_failure/internet	internet	internet	up3ds_failure/internet	internet

Mastercard and Maestro cards use 00, 01, 02, 06, and 07 digit values to indicate the authentication level of the transaction.

Mastercard/Maestro Text Values for ECI Values

ECI Value	Meaning	Mastercard/Maestro
00	Internet, not authenticated	spa/internet
01	Attempted authentication	spa
02	Authenticated	spa
06	Exemption from authentication or network token without 3#D Secure	spa
07	Authenticated merchant-initiated transaction	spa

The payer authentication response contains other information that needs to be passed on for successful authorization. Be sure to include these fields when requesting a separate authorization:

- **consumerAuthenticationInformation.directoryServerTransactionId** (Mastercard, Maestro, UPI only)
- **consumerAuthenticationInformation.eciRaw**
- **consumerAuthenticationInformation.paresStatus**
- **consumerAuthenticationInformation.paSpecificationVersion**
- **consumerAuthenticationInformation.ucafAuthenticationData** (Mastercard/Maestro only)
- **consumerAuthenticationInformation.ucafCollectionIndicator** (Mastercard/Maestro only)
- **consumerAuthenticationInformation.cavv**
- **consumerAuthenticationInformation.xid**

American Express SafeKey

American Express SafeKey is the authentication service in the American Express card network that uses the 3-D Secure protocol to validate customers at checkout. When you request an authorization using a supported card type and a supported processor, you can include payer authentication data in the request.
Before implementing payer authentication for American Express SafeKey, contact customer support to have your account configured for this feature.

Fields Specific to the American Express SafeKey Use Case

These API fields are required specifically for this use case.

consumerAuthenticationInformation.cavv	Required when payer authentication is successful.
processingInformation.commerceIndicator	Set this field to one of these values: • `aesk`: Successful authentication (3#D Secure value of `05`). • `aesk_attempted`: Authentication was attempted (3#D Secure value of `06`). • `internet`: Authentication failed or was not attempted (3#D Secure value of `07`).

Processor-Specific Requirements

Visa Platform Connect

processingInformation.authorizationOptions.transaction	Required only for merchants in Saudi Arabia.

Endpoint

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

Related Information

- *Payer Authentication Developer Guide | REST API*

Required Fields for Processing an Authorization Using American Express SafeKey

These fields must be included in a request for an authorization with American SafeKey. The values for these fields are in the response from the payer authentication validate service. When you request the payer authentication validate and authorization services together, the data is automatically passed from one service to the other.

Important

When using relaxed requirements for address data and the expiration date, not all fields in this list are required. It is your responsibility to determine whether your account is enabled to use this feature and which fields are required. For details about relaxed requirements, see *Relaxed Requirements for Address Data and Expiration Date in Payment Transactions* on page 264.

clientReferenceInformation.code	
consumerAuthenticationInformation.cavv	
consumerAuthenticationInformation.eciRaw	Required when the payer authentication validation service returns a raw unmapped ECI value.
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.commerceIndicator

Set this field to one of these values:

- `aesk`: Successful authentication (3#D Secure value of `05`).
- `aesk_attempted`: Authentication was attempted (3#D Secure value of `06`).
- `internet`: Authentication failed or was not attempted (3#D Secure value of `07`).

Related Information

- *API field reference guide for the REST API*

Optional Field for Processing an Authorization Using American Express SafeKey

This field is optional in a request for an authorization with American Express SafeKey. The value for this field is in the response from the payer authentication validate service. When you request the payer authentication validate and authorization services together, the data is automatically passed from one service to the other.

consumerAuthenticationInformation.xid

Related Information

- *API field reference guide for the REST API*

REST Example: Processing an Authorization Using American Express SafeKey

Endpoint:

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

Request

```
{
  "clientReferenceInformation": {
    "code": "TC50171_3"
  },
  "processingInformation": {
    "commerceIndicator": "aesk"
  },
  "paymentInformation": {
```

```
      "card": {
       "number": "340000XXXXXXX8",
       "expirationMonth": "01",
       "expirationYear": "2025"
     }
    },
     "orderInformation": {
       "amountDetails": {
       "totalAmount": "100",
       "currency": "USD"
     },
     "billTo": {
       "firstName": "John",
       "lastName": "Smith",
       "address1": "201 S. Division St._1",
       "locality": "Foster City",
       "administrativeArea": "CA",
       "postalCode": "94404",
       "country": "US",
       "email": "accept@who.com",
       "phoneNumber": "6504327113"
     }
    },
       "consumerAuthenticationInformation": {
       "cavv": "12345678909876543210ABCDEFabcdefABCDEF123",
       "xid": "12345678909876543210ABCDEFabcdefABCDEF123"
       }
    }
```

Response to a Successful Request

```
{
  "_links": {
    "authReversal": {
      "method": "POST",
      "href": "/pts/v2/payments/6783071542936193303955/reversals"
    },
    "self": {
      "method": "GET",
      "href": "/pts/v2/payments/678307154293619330 3955"
    },
    "capture": {
      "method": "POST",
      "href": "/pts/v2/payments/6783071542936193303955/captures"
    }
  },
  "clientReferenceInformation": {
    "code": "TC50171_3"
  },
  "id": "6783071542936193303955",
  "orderInformation": {
    "amountDetails": {
      "authorizedAmount": "100.00",
      "currency": "USD"
    }
  },
```

```
  "paymentAccountInformation": {
    "card": {
      "type": "003"
    }
  },
  "paymentInformation": {
    "accountFeatures": {
      "currency": "usd",
      "balanceAmount": "70.00"
    },
    "tokenizedCard": {
      "type": "003"
    },
    "card": {
      "type": "003"
    }
  },
  "pointOfSaleInformation": {
    "terminalId": "111111"
  },
  "processorInformation": {
    "approvalCode": "888888",
    "networkTransactionId": "123456789619999",
    "transactionId": "123456789619999",
    "responseCode": "100",
    "avs": {
      "code": "X",
      "codeRaw": "I1"
    }
  },
  "reconciliationId": "62427259FEYR18Q2",
  "status": "AUTHORIZED",
  "submitTimeUtc": "2023-03-08T20:25:54Z"
}
```


JCB J/Secure

JCB J/Secure is the authentication service in the JCB card network that uses the 3#D Secure protocol to validate customers at checkout. When you request an authorization using a supported card type and a supported processor, you can include payer authentication data in the request. The payer authentication services enable you to add payer authentication support to your website without running additional software on your server.

Before implementing payer authentication for JCB J/Secure, contact customer support to have your account configured for this feature.

Fields Specific to the JCB J/Secure Use Case

These API fields are required specifically for this use case.

consumerAuthenticationInformation.cavv	Required when payer authentication is successful.
consumerAuthenticationInformation.xid	Required when payer authentication is successful.
consumerAuthenticationInformation.eciRaw	Required when the payer authentication validation service returns a raw ECI value.
processingInformation.commerceIndicator	Set this field to one of these values: • `js`: Successful authentication for a JCB card (3#D Secure value of `05`). • `js_attempted`: Authentication was attempted for a JCB card (3#D Secure value of `06`). • `js_failure`: or `internet`: Authentication failed or was not attempted for a JCB card (3#D Secure value of `07`).

Endpoint

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

Related Information

- *Payer Authentication Developer Guide | REST API*

Required Fields for Processing an Authorization Using JCB J/Secure Authentication

Use these required fields to process an authorization using JCB J/Secure authentication.

Important

When using relaxed requirements for address data and the expiration date, not all fields in this list are required. It is your responsibility to determine whether your account is enabled to use this feature and which fields are required. For details about relaxed requirements, see *Relaxed Requirements for Address Data and Expiration Date in Payment Transactions* on page 264.

clientReferenceInformation.code

consumerAuthenticationInformation.cavv

consumerAuthenticationInformation.xid

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.commerceIndicator	**Set this field to one of these values:** • `js`: Successful authentication (3#D Secure value of `05`). • `js_attempted`: Authentication was attempted (3#D Secure value of `06`). • `js_failure`: Authentication failed or was not attempted (3#D Secure value of `07`).

Related Information

- *API field reference guide for the REST API*

REST Example: Processing an Authorization Using JCB J/Secure Authentication

Endpoint:

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

Request

```
{
 "clientReferenceInformation": {
   "code": "TC50171_3"
 },
 "processingInformation": {
   "commerceIndicator": "js"
 },
 "paymentInformation": {
   "card": {
```

```
      "number": "340000XXXXXXX8",
      "expirationMonth": "01",
      "expirationYear": "2025"
     }
    },
     "orderInformation": {
      "amountDetails": {
      "totalAmount": "100",
      "currency": "USD"
     },
     "billTo": {
      "firstName": "John",
      "lastName": "Smith",
      "address1": "201 S. Division St._1",
      "locality": "Foster City",
      "administrativeArea": "CA",
      "postalCode": "94404",
      "country": "US",
      "email": "accept@who.com",
      "phoneNumber": "6504327113"
     }
    },
      "consumerAuthenticationInformation": {
      "cavv": "1234567890987654321ABCDEFabcdefABCDEF123",
      "xid": "1234567890987654321ABCDEFabcdefABCDEF123"
      }
    }
```

Response to a Successful Request

```
{
  "_links": {
    "authReversal": {
      "method": "POST",
      "href": "/pts/v2/payments/6783071542936193303955/reversals"
    },
    "self": {
      "method": "GET",
      "href": "/pts/v2/payments/6783071542936193303955"
    },
    "capture": {
      "method": "POST",
      "href": "/pts/v2/payments/6783071542936193303955/captures"
    }
  },
  "clientReferenceInformation": {
    "code": "TC50171_3"
  },
  "id": "6783071542936193303955",
  "orderInformation": {
    "amountDetails": {
      "authorizedAmount": "100.00",
      "currency": "USD"
    }
  },
  "paymentAccountInformation": {
```

```
    "card": {
     "type": "003"
    }
   },
   "paymentInformation": {
    "accountFeatures": {
     "currency": "usd",
     "balanceAmount": "70.00"
    },
    "tokenizedCard": {
     "type": "003"
    },
    "card": {
     "type": "003"
    }
   },
   "pointOfSaleInformation": {
    "terminalId": "111111"
   },
   "processorInformation": {
    "approvalCode": "888888",
    "networkTransactionId": "123456789619999",
    "transactionId": "123456789619999",
    "responseCode": "100",
    "avs": {
     "code": "X",
     "codeRaw": "I1"
    }
   },
   "reconciliationId": "62427259FEYR18Q2",
   "status": "AUTHORIZED",
   "submitTimeUtc": "2023-03-08T20:25:54Z"
}
```


Mastercard Identity Check

Mastercard Identity Check is the authentication service in the Mastercard card network that uses the 3-D Secure protocol in online transactions to authenticate customers at checkout.

Mastercard Identity Check generates a unique, 32-character transaction token, called the account authentication value (AAV) each time a Mastercard Identity Check-enabled account holder makes an online purchase. The AAV binds the account holder to a specific transaction. Mastercard Identity Check transactions use the universal cardholder authentication field (UCAF) as a standard to collect and pass AAV data.

Before implementing payer authentication for Mastercard Identity Check, contact customer support to have your account configured for this feature.

Fields Specific to the Mastercard Identity Check Use Case

These API fields are required specifically for this use case.

consumerAuthenticationInformation.directoryServerTransactionId	Set this field to the transaction ID returned by Mastercard Identity Check during the authentication process.
consumerAuthenticationInformation.paSpecificationVersion	Set this field to the Mastercard Identity Check version returned by Mastercard Identity Check during the authentication process.
consumerAuthenticationInformation.ucafCollectionIndicator	**Set to the last digit of the raw ECI value returned from authentication. For example, if ECI=02, this value should be 2.**
processingInformation.commerceIndicator	**Set this field to one of these values:** • `spa`: Successful authentication (3#D Secure value of `02`). • `spa`: Authentication was attempted (3#D Secure value of `01`). • `spa` or `internet`: Authentication failed or was not attempted (3#D Secure value of `00`)

Endpoint

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

Required Fields for Processing an Authorization Using Mastercard Identity Check

Use these required fields to process an authorization using Mastercard Identity Check.

> **Important**
>
> When using relaxed requirements for address data and the expiration date, not all fields in this list are required. It is your responsibility to determine whether your account is enabled to use this feature and which fields are required. For details about relaxed requirements, see *Relaxed Requirements for Address Data and Expiration Date in Payment Transactions* on page 264.

consumerAuthenticationInformation.directoryServerTransactionId

consumerAuthenticationInformation.paSpecificationVersion

consumerAuthenticationInformation.ucafCollectionIndicator	**Set to the last digit of the raw ECI value returned from authentication. For example, if ECI=02, this value should be 2.**

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	
processingInformation.commerceIndicator	**Set this field to one of these values:** • `spa`: Successful authentication (3#D Secure value of `02`). • `spa`: Authentication was attempted (3#D Secure value of `01`). • `spa` or `internet`: Authentication failed or was not attempted (3#D Secure value of `00`)

Related Information

- *API field reference guide for the REST API*

REST Example: Processing an Authorization Using Mastercard Identity Check

Endpoint:

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

Request

```
{
  "clientReferenceInformation" : {
    "code" : "TC50171_6"
  },
  "consumerAuthenticationInformation" : {
    "ucafCollectionIndicator" : "2",
    "ucafAuthenticationData" : "EHuWW9PiBkWvqE5juRwDzAUFBAk",
    "directoryServerTransactionId" : "f38e6948-5388-41a6-bca4-b49723c19437",
```

```
    "paSpecificationVersion" : "2.2.0"
  },
  "processingInformation" : {
    "commerceIndicator" : "spa"
    },
    "orderInformation" : {
      "billTo" : {
        "country" : "US",
        "lastName" : "Deo",
        "address1" : "201 S. Division St.",
        "postalCode" : "48104-2201",
        "locality" : "Ann Arbor",
        "administrativeArea" : "MI",
        "firstName" : "John",
        "email" : test@cybs.com
      },
      "amountDetails" : {
        "totalAmount" : "105.00",
        "currency" : "USD"
      }
    },
    "paymentInformation" : {
      "card" : {
        "expirationYear" : "2031",
        "number" : "555555555555XXXX",
        "securityCode" : "123",
        "expirationMonth" : "12",
        "type" : "002"
    }
  }
}
```

Response to a Successful Request

```
{
  "_links": {
    "authReversal": {
      "method": "POST",
      "href": "/pts/v2/payments/675899075143665500 4951/reversals"
    },
    "self": {
      "method": "GET",
      "href": "/pts/v2/payments/6758990751436655004951"
    },
    "capture": {
      "method": "POST",
      "href": "/pts/v2/payments/675899075143665500 4951/captures"
    }
  },
  "clientReferenceInformation": {
    "code": "TC50171_3"
  },
  "id": "6758990751436655004951",
  "orderInformation": {
    "amountDetails": {
      "authorizedAmount": "100.00",
```

```
    "currency": "USD"
   }
  },
  "paymentAccountInformation": {
   "card": {
    "type": "002"
   }
  },
  "paymentInformation": {
   "tokenizedCard": {
    "type": "002"
   },
   "card": {
    "type": "002"
   }
  },
  "pointOfSaleInformation": {
   "terminalId": "111111"
  },
  "processorInformation": {
   "approvalCode": "888888",
   "authIndicator": "1",
   "networkTransactionId": "123456789619999",
   "transactionId": "123456789619999",
   "responseCode": "100",
   "avs": {
    "code": "X",
    "codeRaw": "I1"
   }
  },
  "reconciliationId": "71183995FDU0YRTK",
  "status": "AUTHORIZED",
  "submitTimeUtc": "2023-02-08T23:31:15Z"
}
```


Visa Secure

Visa Secure is the authentication service in the Visa card network that uses the 3-D Secure protocol to authenticate customers at checkout. This authentication is a two-step process. First, the cardholder is authenticated by 3-D Secure. Then, the transaction is authorized based on the 3-D Secure evaluation. This section explains how to authorize a card payment based on the 3-D Secure evaluation.
Before implementing Visa Secure, contact customer support to have your account configured for this feature.

Fields Specific to the Visa Secure Use Case

These API fields are required specifically for this use case.

processingInformation.commerceIndicator	Set the value to `vbv` for a successful authentication (3#D Secure value of `05`),

	`vbv_attempted` if authentication was attempted but did not succeed (3#D Secure value of `06`), or `vbv_failure` if authentication failed (3#D Secure value of `07`).
consumerAuthenticationInformation.cavv	Required when payer authentication is successful.

Endpoint

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

Related Information

- *API field reference guide for the REST API*

Required Fields for Processing an Authorization Using Visa Secure

Use these required fields to process an authorization using Visa Secure.

> **Important**
>
> When using relaxed requirements for address data and the expiration date, not all fields in this list are required. It is your responsibility to determine whether your account is enabled to use this feature and which fields are required. For details about relaxed requirements, see *Relaxed Requirements for Address Data and Expiration Date in Payment Transactions* on page 264.

clientReferenceInformation.code	
consumerAuthenticationInformation.cavv	Required when payer authentication is successful. Otherwise, this field is optional.
consumerAuthenticationInformation.xid	
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.commerceIndicator	Set this field to one of these values: • `vbv`: Successful authentication (EMV 3-D Secure value of `05`). • `vbv_attempted`: Authentication was attempted (EMV 3-D Secure value of `06`). • `vbv_failure`: or `internet`: Authentication failed or was not attempted (EMV 3-D Secure value of `07`).

Related Information

- *API field reference guide for the REST API*

Related information
- API field reference guide for the REST API

REST Example: Validating and Authorizing a Transaction

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

```
{
 "clientReferenceInformation": {
  "code": "TC50171_3"
 },
 "processingInformation": {
  "commerceIndicator": "vbv"
 },
 "paymentInformation": {
  "card": {
   "number": "41111111XXXXXXX1",
   "expirationMonth": "01",
   "expirationYear": "2026"
  }
 },
 "orderInformation": {
  "amountDetails": {
   "totalAmount": "100",
   "currency": "USD"
  },
```

```
    "billTo": {
     "firstName": "John",
     "lastName": "Smith",
     "address1": "201 S. Division St._1",
     "locality": "Foster City",
     "administrativeArea": "CA",
     "postalCode": "94404",
     "country": "US",
     "email": "accept@who.com",
     "phoneNumber": "6504327113"
    }
   },
   "consumerAuthenticationInformation": {
   "cavv": "12345678909876543 21ABCDEFabcdefABCDEF123",
    "xid": "12345678909876543 21ABCDEFabcdefABCDEF123"
   }
}
```

Response

```
{
  "_links": {
    "authReversal": {
      "method": "POST",
      "href": "/pts/v2/payments/6758954108726900304951/reversals"
    },
    "self": {
      "method": "GET",
      "href": "/pts/v2/payments/675895410872690030 4951"
    },
    "capture": {
      "method": "POST",
      "href": "/pts/v2/payments/6758954108726900304951/captures"
    }
  },
  "clientReferenceInformation": {
    "code": "TC50171_3"
  },
  "id": "6758954108726900304951",
  "orderInformation": {
    "amountDetails": {
      "authorizedAmount": "100.00",
      "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": "711764833DU1FCQD",
   "status": "AUTHORIZED",
   "submitTimeUtc": "2023-02-08T22:30:11Z"
}
```

Relaxed Requirements for Address Data and Expiration Date in Payment Transactions

With relaxed requirements for address data and the expiration date, not all standard payment request fields are required. It is your responsibility to determine whether your account is enabled to use this feature and which fields are required.

Requirements

You must contact customer support in order to enable relaxed requirements for address data and expiration date.

Services

Relaxed requirements for address data and expiration date are supported for these services:

- Authorization
- Capture
- Stand-alone credit
- Subscription create
- Subscription update

Relaxed Fields

Important

When relaxed requirements for address data and expiration date are enabled for your Cybersource account, and your service request does not include one or more of the fields in the following list, you increase the risk of declined transactions and fraud depending on your location, your processor, and the cardholder's issuing bank.

It is your responsibility to determine whether a field is required for the transaction you are requesting. For example, an issuing bank can decline an authorization request for a recurring transaction with a Visa Europe card if the expiration date is incorrect, invalid, or missing. If you do not provide the correct expiration date for a recurring transaction the authorization request may be declined.

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	**When you include this field in your request, you must also include paymentInformation.card.expirationYear.** **You can submit an expiration date that has expired. This exception does not apply when you combine any of the services listed above with any other service.** **This field is required for payment network token transactions and subscription creation requests.**
paymentInformation.card.expirationYear	**When you include this field in your request, you must also include paymentInformation.card.expirationMonth.** **You can submit an expiration date that has expired. This exception does not apply**

when you combine any of the services listed above with any other service.

This field is required for payment network token transactions and subscription creation requests.

Split Shipments Processing

Split shipments enable you to split an order into multiple shipments with multiple captures. You can use this feature when a customer orders a product that is not yet available.

> **Important**
>
> Split shipments are not available for Mastercard transactions in the IDR currency on Visa Platform Connect.

Multiple partial captures and split shipments are not the same feature. The processor provides the multiple partial captures feature, while Cybersource provides the split shipment feature.

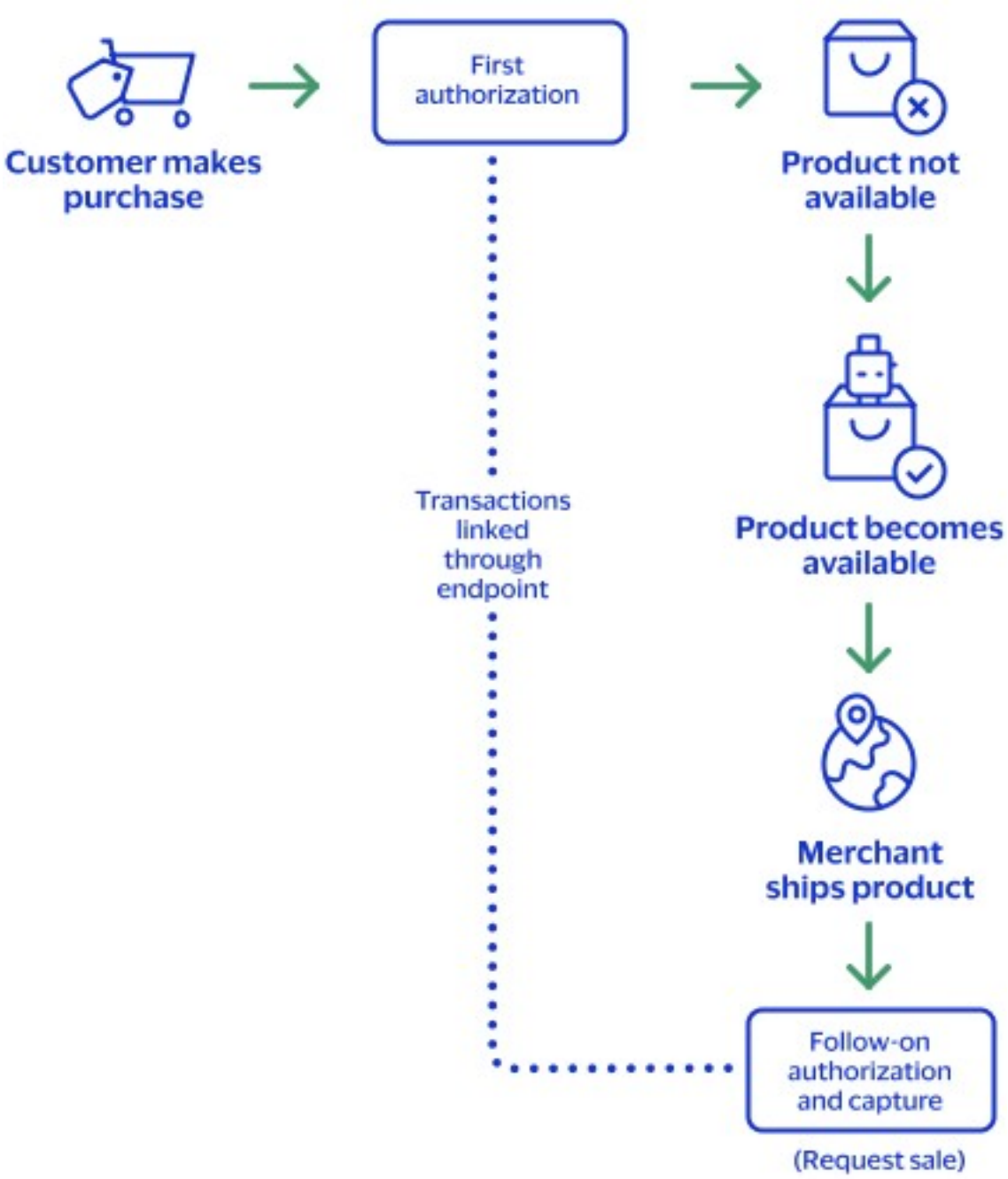
Requirements for Using Split Shipments

The requirements for using split shipments are you must use Visa Platform Connect and contact customer support to have your account configured for this feature.

> **Important**
>
> A Visa Platform Connect account can only be enabled for either the multiple partial captures or split shipments feature, but not both.

Authorizing a Sale for a Product Not Yet Available

When the customer purchases a product that is not yet available, you can request an authorization and a sale. First request an authorization to ensure that funds are available. After the product becomes available, ship the product and request a sale. Cybersource then links the follow-on authorization to the first authorization, and then links to the capture request.



Authorizing a Sale for a Product not yet Available

Step 1: Requesting an authorization
Request an authorization to ensure that funds are available before the product is available for immediate shipment. The authorization request requires no additional fields or requirements than a basic authorization.
Step 2: Processing a sale
When the product becomes available, ship the product and request a sale. The follow-on authorization requires you to submit a sale request that includes the

processingInformation.linkId field in addition to the basic fields required for every sale request. The **processingInformation.linkId** field in an authorization request triggers the split-shipment functionality.
Set the **processingInformation.linkId** field to the **{id}** value from the endpoint.
Field Specific to authorizing a sale for a product not yet available:
First Authorization Response: The **{id}** value is returned in the endpoint.
Follow-on Authorization Request: `processingInformation.linkId=SWVdPS5IM`
Step 3: Cybersource attempts to link the follow-on authorization request to the first authorization

- If the **processingInformation.linkId** value is valid, the follow-on authorization is linked to the original authorization in the Business Center and in reports.
- If the **processingInformation.linkId** value is not valid, the follow-on authorization is not linked to the original authorization in the Business Center and in reports.

Step 4: Cybersource links the capture request

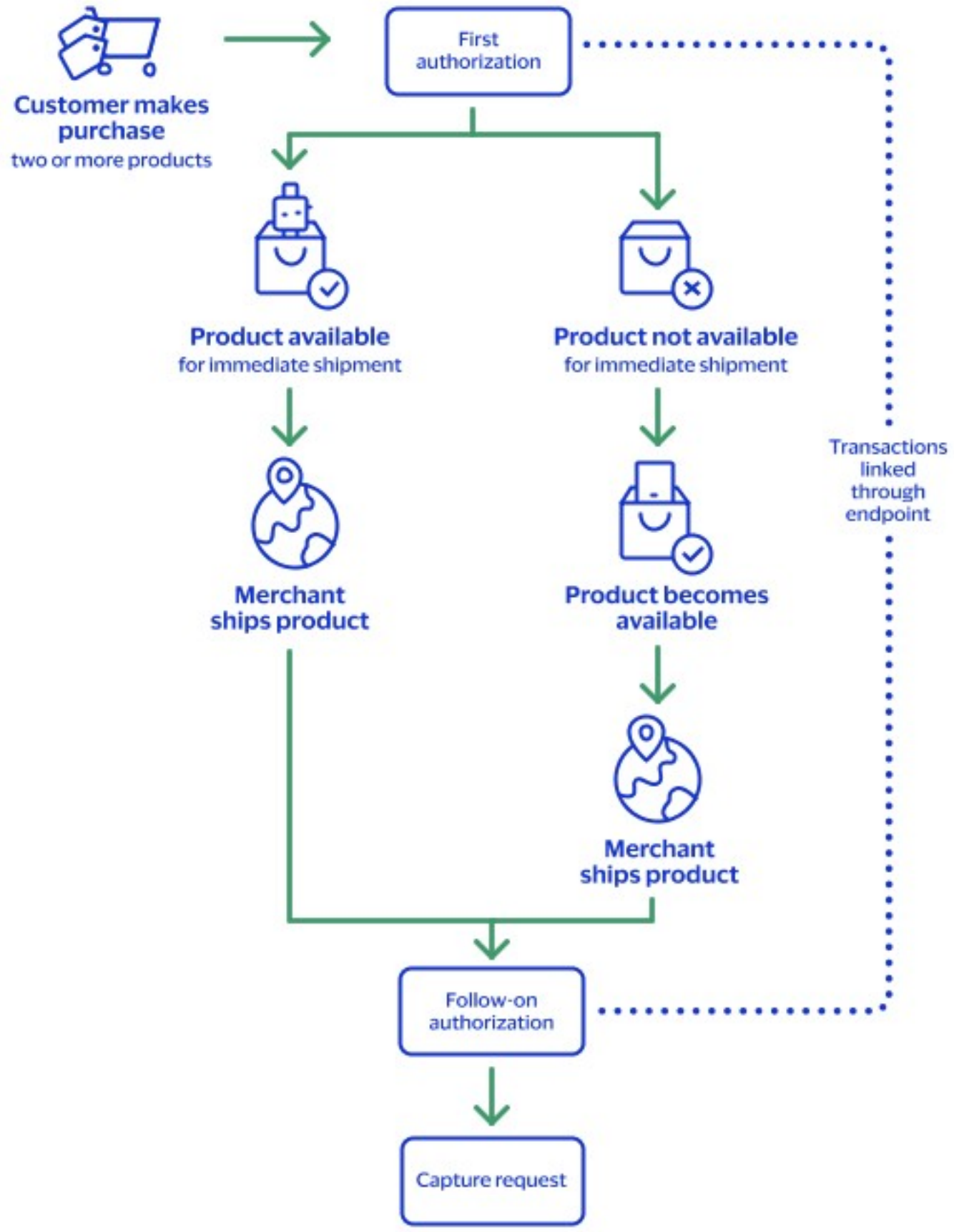
- If the **processingInformation.linkId** value for the follow-on authorization was valid, all three transactions (first authorization, follow-on authorization, capture) are linked together in the Business Center and in reports.
- If the **processingInformation.linkId** value for the follow-on authorization was not valid, the second authorization and capture are linked to each other in the Business Center and in reports, but they are not linked to the first authorization.

Related Information

- See *Basic Authorizations* on page 50 for information on how to process a basic authorization.
- See *Sales* on page 106 for information on how to process a sale.

Processing Two Authorizations and a Capture for Multiple Products

When the customer purchases a product that is not yet available, you can request two authorizations and a capture. First request an authorization to ensure that funds are available, and then ship the available products. After the remaining products become available, request follow-on authorization to ensure funds are still available. Ship the remaining products, and request a capture. Cybersource links the follow-on authorization to the first authorization and the capture request to the other transactions.



Processing Two Authorizations and a Capture for Multiple Products

Step 1: Requesting an authorization

Request an authorization to ensure that funds are available for one or more of the products that are available for immediate shipment. The authorization request requires no additional fields or requirements than a basic authorization.
Step 2: Requesting a follow-on authorization
After the product becomes available, request a follow-on authorization to ensure that funds are still available. The follow-on authorization request must include the **processingInformation.linkId** field in addition to the basic fields required for every authorization request. The **processingInformation.linkId** field in an authorization request triggers the split shipment functionality.
Set the **processingInformation.linkId** field to the **{id}** value from the endpoint.
Field specific to requesting a follow-on authorization request:
First Authorization Response: The **{id}** value is returned in the endpoint.
Follow-on Authorization Request: `processingInformation.linkId=SWVdPS5IM`
Step 3: Cybersource attempts to link the follow-on authorization request to the first authorization

- If the **processingInformation.linkId** value is valid, the follow-on authorization is linked to the original authorization in the Business Center and in reports.
- If the **processingInformation.linkId** value is not valid, the follow-on authorization is not linked to the original authorization in the Business Center and in reports.

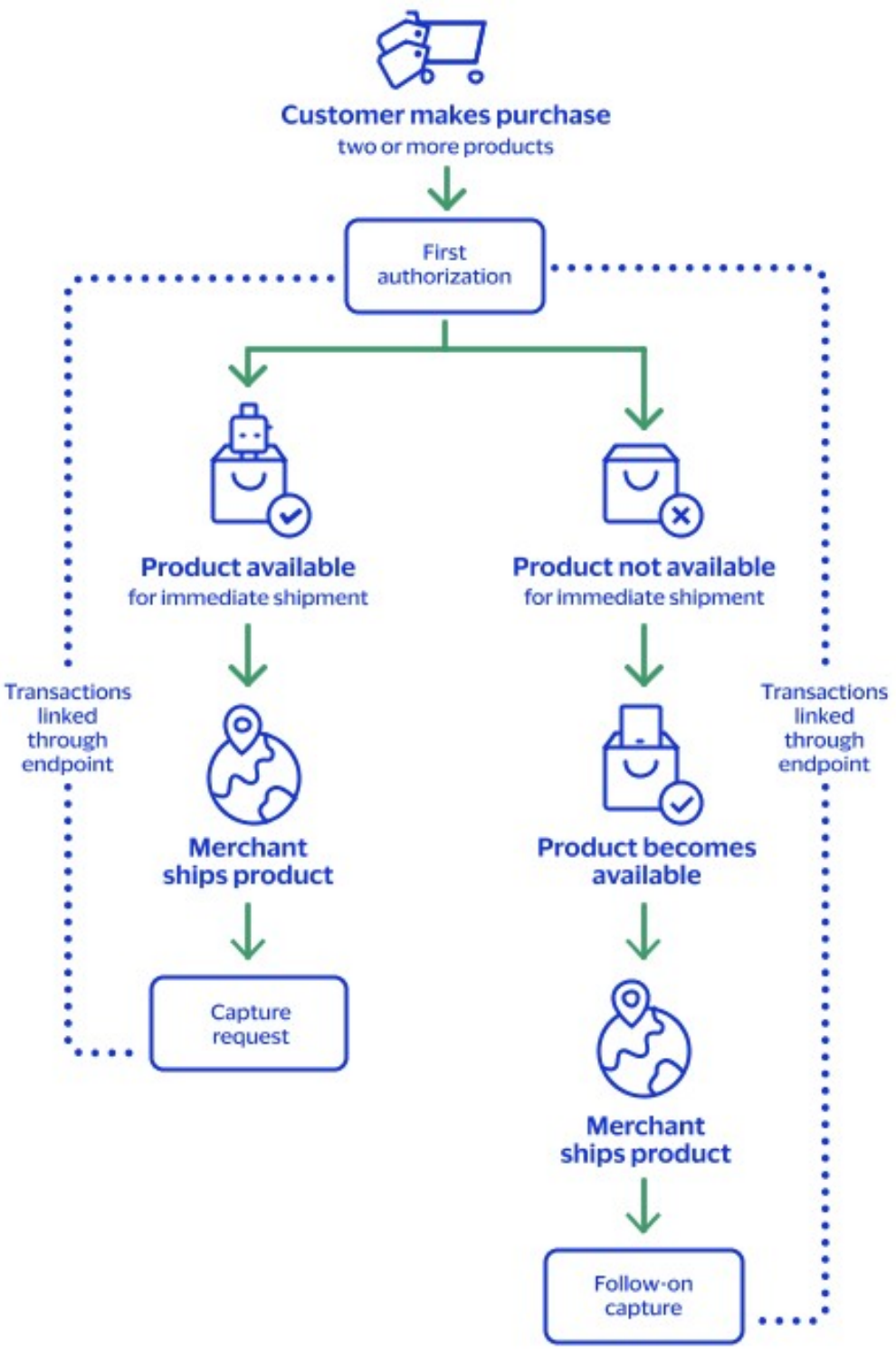
Step 4: Requesting a capture
You ship the product and request a capture. The capture request requires only the basic fields as any capture request.
Step 5: Cybersource attempts to link the capture request to the other transactions
All three transactions (first authorization, follow-on authorization, capture) are linked together in the Business Center and in reports.

Related Information

- See *Basic Authorizations* on page 50 for information on how to process a basic authorization.
- See *Captures* on page 93 for information on how to process a capture.

Processing an Authorization and Two Captures for Multiple Products

When the customer orders multiple products and one is not available, you must request an authorization to ensure funds are available. You ship the products that are available and request a capture for the amount of the shipped products. When the remaining product becomes available, ship the product and request a follow-on capture for the amount of the product. Cybersource performs a system-generated authorization for the follow-on capture request. Cybersource then links the capture request. You receive the status of the follow-on capture request and its associated system-generated authorization.



Processing an Authorization and Two Captures for Multiple Products

Step 1: Requesting an authorization

Request an authorization to ensure that funds are available for one or more products that are available for immediate shipment. The authorization request requires no additional fields or requirements other than a basic authorization.
Step 2: Requesting a capture
Ship the available product and request a capture while you wait for the remaining product to become available. The capture request requires only the basic fields as any capture request.
Step 3: Requesting a follow-on capture
When the remaining product becomes available, ship it and request a capture for that amount. The capture request requires only the basic fields as any capture request.
Step 4: Cybersource performs a system-generated authorization
Cybersource performs a system-generated authorization for the follow-on capture request and link it to the original authorization in the Business Center and in reports. Cybersource processes the capture request as a split shipment request because your account is already enabled for split shipments.
Step 5: Cybersource attempts to link the capture request to the other transactions
The capture is linked to the authorizations in the Business Center and in reports through the request IDs as with any capture. All four transactions (first authorization, system-generated authorization, first capture, follow-on capture) are linked together in the Business Center and in reports.
Step 6: Cybersource provides the status
The status of the follow-on capture request and its associated system-generated authorization becomes available.

Related Information

- See *Basic Authorizations* on page 50 for information on how to process a basic authorization.
- See *Captures* on page 93 for information on how to process a capture.

Processing Payments Using Credentials

This section provides the information you need in order to process payments using credentials.

Additional Resources for Credentialed Transactions

For more information, see these guides:

- *Credentialed Transactions Developer Guide*
- *Token Management Service Developer Guide*
- *API field reference guide for the REST API*
- Github repositories: *https://github.com/Cybersource?q=rest-sample&type=all&language=&sort=*

Customer-Initiated Transactions with Credentials on File

A customer-initiated transaction (CIT) is a transaction initiated by the customer. There are two types of CITs:

- Customer transactions during which the credentials are stored for future customer-initiated transactions
- Customer transactions during which the credentials are stored for future merchant-initiated transactions

Customers can initiate a CIT at a merchant payment terminal, through an online purchase transaction, or by making a purchase using a previously stored credential.

Business Center

You can create a new customer-initiated transaction in the Business Center by going to the One-Time Payments section and requesting a new authorization. When you have entered the customer's information, you can store the customer's credentials with the customer's permission in the Payment Information section. By doing so, you can perform merchant-initiated transactions for payments that the customer has pre-approved. For more information on how to perform a MIT in the Business Center, see *Merchant-Initiated No-Show Transactions with PAN* on page 311.

Storing Customer Credentials with a CIT and PAN

Before you can perform a merchant-initiated transaction (MIT) or a customer-initiated transaction (CIT) with credentials-on-file (COF), you must store the customer's credentials for later use. Further, before you can store the user's credentials, you must get the customer's consent to store their private information. This is also known as establishing a relationship with the customer.

Endpoint

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

Required Fields for Storing Customer Credentials During a CIT

Use these required fields for storing customer credentials during a customer-initiated transaction.

> **Important**
>
> When using relaxed requirements for address data and the expiration date, not all fields in this list are required. It is your responsibility to determine whether your account is enabled to use this feature and which fields are required. For details about relaxed requirements, see *Relaxed Requirements for Address Data and Expiration Date in Payment Transactions* on page 264.

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.phoneNumber

orderInformation.billTo.postalCode

paymentInformation.card.expirationMonth

paymentInformation.card.expirationYear

paymentInformation.card.number

processingInformation.authorizationOptions.initiator. credentialStoredOnFile **Set the value to** `true`.

REST Example: Storing Customer Credentials During a CIT

Endpoint:

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

Request

```
{
  "processingInformation": {
    "authorizationOptions": {
      "initiator": {
        "credentialStoredOnFile": "true"
      }
    }
  },
  "orderInformation": {
    "billTo": {
      "firstName": "John",
      "lastName": "Doe",
      "address1": "201 S. Division St.",
      "postalCode": "48104-2201",
      "locality": "Ann Arbor",
      "administrativeArea": "MI",
      "country": "US",
      "email": "test@cybs.com",
      "phoneNumber": "5554327113"
    },
    "amountDetails": {
      "totalAmount": "100.00",
      "currency": "USD"
    }
  },
  "paymentInformation": {
    "card": {
      "expirationYear": "2031",
      "number": "4111xxxxxxxxxxx",
```

```
        "expirationMonth": "12"
      }
    }
}
```

Response to a Successful Request

```
{
  "_links": {
    "authReversal": {
      "method": "POST",
      "href": "/pts/v2/payments/652818719894607630300 4/reversals"
    },
    "self": {
      "method": "GET",
      "href": "/pts/v2/payments/65281871989460763030 04"
    },
    "capture": {
      "method": "POST",
      "href": "/pts/v2/payments/652818719894607630300 4/captures"
    }
  },
  "clientReferenceInformation": {
    "code": "1652818719876"
  },
  "id": "652818719894607630300 4",
  "orderInformation": {
    "amountDetails": {
      "authorizedAmount": "100.00",
      "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": "63165088Z3AHV91G",
  "status": "AUTHORIZED",
  "submitTimeUtc": "2022-05-17T20:18:40Z"
}
```

Storing Customer Credentials with a CIT and TMS

Before you can perform a merchant-initiated transaction (MIT) or a customer-initiated transaction (CIT) with credentials-on-file (COF), you must store the customer's credentials for later use. Further, before you can store the user's credentials, you must get the customer's consent to store their private information. This is also known as establishing a relationship with the customer.

Creating a TMS Token

When sending the initial CIT, you can create a TMS token to store the customer's credentials for the subsequent MITs. To create a TMS token, include the **processingInformation.actionTokenTypes** field in the authorization request. Set the field to one of these values based on the TMS token type you want to create:

Customer	**Customer tokens store one or more customer payment instrument tokens and shipping address tokens.** **Including a customer token in subsequent MITs eliminates the need to include billing information, card information, and the previous transaction's ID.** `"processingInformation": { "actionTokenTypes": [ "customer" ]` **For more information about this TMS token type, see *Customer Tokens* in the Token Management Service Developer Guide.**
Payment Instrument	**Payment instrument tokens store an instrument identifier token, card information, and billing information. Payment instruments are not linked to a customer token. Including a payment instrument in subsequent MITs eliminates the need to include billing information, card**

	information, and the previous transaction's ID. `"processingInformation": { "actionTokenTypes": [ "paymentInstrument" ]` **For more information about this TMS token type, see *Payment Instrument Token* in the Token Management Service Developer Guide.**
Instrument Identifier	**Instrument identifier tokens store a PAN. Including an instrument identifier in subsequent MITs eliminates the need to include a PAN and the previous transaction's ID.** `"processingInformation": { "actionTokenTypes": [ "instrumentIdentifier" ]` **For more information about this TMS token type, see *Instrument Identifier Token* in the Token Management Service Developer Guide.**
Instrument Identifier, Payment Instrument, and Customer Identifier	**You can also create multiple TMS token types in the same authorization. This example includes an instrument identifier, a payment instrument, and a customer token in the same authorization:** `"processingInformation": { "actionTokenTypes": [ "instrumentIdentifier", "paymentInstrument", "customer" ]`

Endpoint

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

Required Fields for Storing a Customers Credentials with a CIT and TMS

> **Important**
>
> When using relaxed requirements for address data and the expiration date, not all fields in this list are required. It is your responsibility to determine whether your account is enabled to use this feature and which fields are required. For details about relaxed requirements, see *Relaxed Requirements for Address Data and Expiration Date in Payment Transactions* on page 264.

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.phoneNumber

orderInformation.billTo.postalCode

paymentInformation.card.expirationMonth

paymentInformation.card.expirationYear

paymentInformation.card.number

processingInformation.actionList

Set the value to `TOKEN_CREATE`

processingInformation.actionTokenTypes

Set to one or more of these values:

- `customer`
- `instrumentIdentifier`
- `paymnentInstrument`

Example: Storing a Customer's Credentials with a CIT and TMS

Use this example as a reference for how to store a customer's credentials in a TMS token within a CIT.

Endpoint:

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

Request

```
{
 "processingInformation": {
  "actionList": [
   "TOKEN_CREATE"
  ],
  "actionTokenTypes": [
   "instrumentIdentifier"
  ]
 },
 "paymentInformation": {
  "card": {
   "number": "4111111111111111",
   "expirationMonth": "12",
   "expirationYear": "2031",
   "securityCode": "123"
  }
 },
 "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"
  }
 }
}
```

Response to a Successful Request

```
{
 "_links": {
  "authReversal": {
   "method": "POST",
   "href": "/pts/v2/payments/6972267090226779103955/reversals"
  },
  "self": {
   "method": "GET",
   "href": "/pts/v2/payments/697226709022677910 3955"
  },
  "capture": {
   "method": "POST",
   "href": "/pts/v2/payments/6972267090226779103955/captures"
  }
 },
 "clientReferenceInformation": {
```

```
    "code": "TC50171_3"
  },
  "id": "6972267090226779103955",
  "orderInformation": {
   "amountDetails": {
    "authorizedAmount": "102.21",
    "currency": "USD"
   }
  },
  "paymentAccountInformation": {
   "card": {
    "type": "001"
   }
  },
  "paymentInformation": {
   "tokenizedCard": {
    "type": "001"
   },
   "card": {
    "type": "001"
   }
  },
  "pointOfSaleInformation": {
   "terminalId": "111111"
  },
  "processorInformation": {
   "paymentAccountReferenceNumber": "V0010013022298169667504231315",
   "approvalCode": "888888",
   "networkTransactionId": "123456789619999",
   "transactionId": "123456789619999",
   "responseCode": "100",
   "avs": {
    "code": "X",
    "codeRaw": "I1"
   }
  },
  "reconciliationId": "62506622XNMR6Q1Y",
  "status": "AUTHORIZED",
  "submitTimeUtc": "2023-10-13T19:51:49Z",
  "tokenInformation": {
   "instrumentidentifierNew": false,
   "instrumentIdentifier": {
    "state": "ACTIVE",
    "id": "7010000000016241111"
   }
  }
}
```

Using Stored Customer Credentials During a CIT

After customers store their credentials on file, they can recall these credentials to use with subsequent transactions.

Endpoint

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

Required Fields for Retrieving Customer Credentials During a Customer-Initiated Transaction

Use these required fields to retrieve customer credentials during a customer-initiated transaction.

Important

When using relaxed requirements for address data and the expiration date, not all fields in this list are required. It is your responsibility to determine whether your account is enabled to use this feature and which fields are required. For details about relaxed requirements, see *Relaxed Requirements for Address Data and Expiration Date in Payment Transactions* on page 264.

orderInformation.amountDetails.currency

orderInformation.amountDetails.totalAmount

orderInformation.billTo.address1

orderInformation.billTo.administrativeArea

orderInformation.billTo.country

orderInformation.billTo.email

orderInformation.billTo.lastName

orderInformation.billTo.locality

orderInformation.billTo.phoneNumber

orderInformation.billTo.postalCode

paymentInformation.card.expirationMonth

paymentInformation.card.expirationYear

paymentInformation.card.number

processingInformation.authorizationOptions.initiator. storedCredentialUsed	**Set field to** `true`**.**

Card-Specific Required Field for Retrieving Customer Credentials During a CIT

Some card companies require additional information when making authorizations with stored credentials.

Discover

Discover requires the authorization amount from the original transaction when sending a request:

processingInformation.authorizationOptions.initiator.merchantInitiatedTransaction. originalAuthorizedAmount

REST Example: Retrieving Customer Credentials During a CIT

Endpoint:

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

Request

```
{
  "processingInformation": {
    "authorizationOptions": {
      "initiator": {
        "storedCredentialUsed": "true"
      }
    }
  },
  "orderInformation": {
    "billTo": {
      "firstName": "John",
      "lastName": "Doe",
      "address1": "201 S. Division St.",
      "postalCode": "48104-2201",
      "locality": "Ann Arbor",
      "administrativeArea": "MI",
      "country": "US",
      "email": "test@cybs.com",
      "phoneNumber": "5554327113"
    },
    "amountDetails": {
      "totalAmount": "100.00",
      "currency": "USD",
      "originalAmount": "100"
        // Discover card Only
    }
  },
  "paymentInformation": {
```

```
      "card": {
        "expirationYear": "2031",
        "number": "4111xxxxxxxxxxx",
        "expirationMonth": "12"
      }
    },
   "processorInformation": {
    "transactionId": "123456789610000"
   }
}
```

Response to a Successful Request

```
},
   "paymentAccountInformation": {
     "card": {
       "type": "002"
     }
   },
   "paymentInformation": {
     "tokenizedCard": {
       "type": "002"
     },
     "card": {
       "type": "002"
     }
   },
   "pointOfSaleInformation": {
     "terminalId": "111111"
   },
   "processorInformation": {
     "approvalCode": "888888",
     "authIndicator": "1",
     "networkTransactionId": "123456789619999",
     "transactionId": "123456789619999",
     "responseCode": "100",
     "avs": {
       "code": "X",
       "codeRaw": "I1"
     }
   },
   "reconciliationId": "63740353A3AJ2NSH",
   "status": "AUTHORIZED",
   "submitTimeUtc": "2022-05-20T19:13:06Z"
}
```

Merchant-Initiated Delayed Transaction with PAN

Delayed charge transaction is performed to process a supplemental account charge after original services have been rendered and respective payment has been processed.

Endpoint

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

Required Fields for Processing a Merchant-Initiated Delayed Transaction

Use these required fields to process a merchant-initiated delayed transaction.

Important

When using relaxed requirements for address data and the expiration date, not all fields in this list are required. It is your responsibility to determine whether your account is enabled to use this feature and which fields are required. For details about relaxed requirements, see *Relaxed Requirements for Address Data and Expiration Date in Payment Transactions* on page 264.

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.phoneNumber

orderInformation.billTo.postalCode

paymentInformation.card.expirationMonth

paymentInformation.card.expirationYear

paymentInformation.card.number

processorInformation.cardReferenceData	**Required only for token transactions with Discover or Diners Club. Set this field to the processorInformation.cardReferenceData field that was in the response message when you obtained the customer's credentials.**
processingInformation. authorizationOptions.initiator.	For Discover and American Express cards, use the transaction ID from the

merchantInitiatedTransaction. previousTransactionId	original transaction. For Visa, use the last successful transaction ID.
processingInformation.authorizationOptions.initiator. merchantInitiatedTransaction.reason	**Set the value to** 2**.**
processingInformation. authorizationOptions. initiator. type	**Set the value to** merchant**.**
issuerInformation.transactionInformation	**Required only for token transactions with Discover or Diners Club. Set this field to the processorInformation.transactionID field that was in the response message when you obtained the customer's credentials.**

Card-Specific Required Field for Processing a Merchant-Initiated Transactions

Discover

The listed card requires an additional field:

processingInformation. authorizationOptions. initiator. merchantInitiatedTransaction. originalAuthorizedAmount	**Provide the original transaction amount.**

REST Example: Processing a Merchant-Initiated Delayed Authorization Transaction

Endpoint:

- Production: POST https://api.cybersource.com/pts/v2/payments
- Test: POST https://apitest.cybersource.com/pts/v2/payments

Request

```
{
   "orderInformation": {
  "billTo" : {
    "country" : "US",
    "lastName" : "Kim",
    "address1" : "201 S. Division St.",
    "postalCode" : "48104-2201",
    "locality" : "Ann Arbor",
    "administrativeArea" : "MI",
    "firstName" : "Kyong-Jin",
        "phoneNumber": "5554327113",
    "email" : "test@cybs.com"
   },
     "amountDetails": {
        "totalAmount": "120.00",
        "currency": "USD"
```

```
      }
   },
   "paymentInformation": {
      "card": {
         "expirationYear": "2031",
         "number": "4111xxxxxxxxxxx",
         "expirationMonth": "12"
      }
   },
   "processingInformation": {
      "authorizationOptions": {
         "initiator": {
       "type": "merchant",
          "merchantInitiatedTransaction": {
           "originalAuthorizedAmount": "100",
              // Discover only
           "previousTransactionId": "123456789619999",
           "reason": "2"
          }
         }
      }
   }
}
```

Response to a Successful Request

```
{
   "_links": {
      "authReversal": {
         "method": "POST",
         "href": "/pts/v2/payments/6534213653516599003001/reversals"
      },
      "self": {
         "method": "GET",
         "href": "/pts/v2/payments/653421365351659900301"
      },
      "capture": {
         "method": "POST",
         "href": "/pts/v2/payments/6534213653516599003001/captures"
      }
   },
   "clientReferenceInformation": {
      "code": "1653421365327"
   },
   "id": "653421365351659900301",
   "orderInformation": {
      "amountDetails": {
         "authorizedAmount": "120.00",
         "currency": "USD"
      }
   },
   "paymentAccountInformation": {
      "card": {
         "type": "002"
      }
   },
```

```
    "paymentInformation": {
      "tokenizedCard": {
        "type": "002"
      },
      "card": {
        "type": "002"
      }
    },
    "pointOfSaleInformation": {
      "terminalId": "111111"
    },
    "processorInformation": {
      "approvalCode": "888888",
      "authIndicator": "1",
      "networkTransactionId": "123456789619999",
      "transactionId": "123456789619999",
      "responseCode": "100",
      "avs": {
        "code": "X",
        "codeRaw": "I1"
      }
    },
    "reconciliationId": "64365475T3K10Q1D",
    "status": "AUTHORIZED",
    "submitTimeUtc": "2022-05-24T19:42:45Z"
}
```


Merchant-Initiated Delayed Transaction with TMS

Delayed charge transaction is performed to process a supplemental account charge after original services have been rendered and respective payment has been processed. Delayed charges are typical for lodging transactions and auto rental transactions.
This section describes how to process a merchant-initiated delayed transaction using these TMS token types:

Customer	Customer tokens store one or more customer payment instrument tokens and shipping address tokens. Including a customer token eliminates the need to include billing information, card information, and the previous transaction's ID.


```
"paymentInformation": {
 "customer": {
  "id": "07C9CA98022DA498E063A2598D0AA400"
 }
```

```
}
```

For more information about this TMS token type, see *Customer Tokens* in the Token Management Service Developer Guide.

Payment Instrument

Payment instrument tokens store an instrument identifier token, card information, and billing information. Payment instruments are not linked to a customer token.

Including a payment instrument eliminates the need to include billing information, card information, and the previous transaction's ID.

```
"paymentInformation": {
 "paymentInstrument": {
  "id": "07CA24EF20F9E2C9E063A2598D0A8565"
 }
}
```

For more information about this TMS token type, see *Payment Instrument Token* in the Token Management Service Developer Guide.

Instrument Identifier

Instrument identifier tokens store only a PAN. Including an instrument identifier eliminates the need to include a PAN and the previous transaction's ID.

```
"paymentInformation": {
 "instrumentIdentifier": {
  "id": "7010000000016241111"
 }
}
```

For more information about this TMS token type, see *Instrument Identifier Token* in the Token Management Service Developer Guide.

Endpoint

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

Required Fields for MIT Delayed Transaction with TMS

Include these Required Fields

> **Important**
>
> When using relaxed requirements for address data and the expiration date, not all fields in this list are required. It is your responsibility to determine whether your account is enabled to use this feature and which fields are required. For details about relaxed requirements, see *Relaxed Requirements for Address Data and Expiration Date in Payment Transactions* on page 264.

orderInformation.amountDetails.currency

orderInformation.amountDetails.totalAmount

paymentInformation.[tokentype].id	Where **[tokentype]** is the TMS token type you are using: • **customer** • **instrumentIdentifier** • **paymentInstrument**
processingInformation. authorizationOptions. initiator. merchantInitiatedTransaction. reason	Set the value to 2.

Instrument Identifier Required Fields

If you are using the **paymentInformation.instrumentIdentifier.id** token, include these required fields in addition to the required fields listed above.

orderInformation.billTo.address1

orderInformation.billTo.administrativeArea

orderInformation.billTo.country

orderInformation.billTo.email

orderInformation.billTo.firstName

orderInformation.billTo.lastName

orderInformation.billTo.locality

orderInformation.billTo.phoneNumber

orderInformation.billTo.postalCode

paymentInformation.card.expirationMonth

paymentInformation.card.expirationYear

Card-Specific Fields

Include these fields when processing an authorization with these card types.
The listed card type requires an additional field.

Diners Club	**processorInformation.cardReferenceData:** **Required only for token transactions. Set this field to the processorInformation.cardReferenceData field that was in the response message when you obtained the customer's credentials.** **issuerInformation.transactionInformation:** **Required only for token transactions. Set this field to the processorInformation.transactionID field that was in the response message when you obtained the customer's credentials.**
Discover	**processingInformation.authorizationOptions.initiator.merchantInitiatedTransaction.originalAuthorizedAmou** **Set to the original transaction amount.** **processorInformation.cardReferenceData** **Required only for token transactions. Set this field to the processorInformation.cardReferenceData field that was in the response message when you obtained the customer's credentials.** **issuerInformation.transactionInformation** **Required only for token transactions. Set this field to the processorInformation.transactionID field that was in the response message when you obtained the customer's credentials.**

Example: MIT Delayed Transaction with TMS Instrument Identifier

Endpoint:

- Production: POST https://api.cybersource.com/pts/v2/payments
- Test: POST https://apitest.cybersource.com/pts/v2/payments

Request

```
{
  "processingInformation": {
    "authorizationOptions": {
```

```
    "initiator": {
     "merchantInitiatedTransaction": {
      "reason": "2"
     }
    }
   }
  },
  "paymentInformation": {
   "card": {
    "expirationMonth": "12",
    "expirationYear": "2031"
   },
   "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"
   }
  }
}
```

Response to a Successful Request

```
{
 "_links": {
  "authReversal": {
   "method": "POST",
   "href": "/pts/v2/payments/6976922830456934003954/reversals"
  },
  "self": {
   "method": "GET",
   "href": "/pts/v2/payments/697692283045693400 3954"
  },
  "capture": {
   "method": "POST",
   "href": "/pts/v2/payments/6976922830456934003954/captures"
  }
 },
 "clientReferenceInformation": {
  "code": "1697692283160"
 },
 "id": "6976922830456934003954",
```

```
  "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"
  },
  "processingInformation": {
   "paymentSolution": "015"
  },
  "processorInformation": {
   "paymentAccountReferenceNumber": "V0010013022298169667504231315",
   "approvalCode": "888888",
   "networkTransactionId": "123456789619999",
   "transactionId": "123456789619999",
   "responseCode": "100",
   "avs": {
    "code": "X",
    "codeRaw": "I1"
   }
  },
  "reconciliationId": "62700184NNMR6XFK",
  "status": "AUTHORIZED",
  "submitTimeUtc": "2023-10-19T05:11:23Z"
}
```


Example: MIT Delayed Transaction with TMS Payment Instrument

Endpoint:

- Production: POST https://api.cybersource.com/pts/v2/payments
- Test: POST https://apitest.cybersource.com/pts/v2/payments

Request


```
{
```

```
  "processingInformation": {
   "authorizationOptions": {
    "initiator": {
     "merchantInitiatedTransaction": {
      "reason": "2"
     }
    }
   }
  },
  "paymentInformation": {
   "paymentInstrument": {
    "id": "080AE120369A7947E063A2598D0A718F"
   }
  },
  "orderInformation": {
   "amountDetails": {
    "totalAmount": "102.21",
    "currency": "USD"
   }
  }
}
```

Response to a Successful Request

```
{
  "_links": {
   "authReversal": {
    "method": "POST",
    "href": "/pts/v2/payments/6976917718796256603955/reversals"
   },
   "self": {
    "method": "GET",
    "href": "/pts/v2/payments/697691771879625660 3955"
   },
   "capture": {
    "method": "POST",
    "href": "/pts/v2/payments/6976917718796256603955/captures"
   }
  },
  "clientReferenceInformation": {
   "code": "1697691771976"
  },
  "id": "6976917718796256603955",
  "orderInformation": {
   "amountDetails": {
    "authorizedAmount": "102.21",
    "currency": "USD"
   }
  },
  "paymentAccountInformation": {
   "card": {
    "type": "001"
   }
  },
  "paymentInformation": {
   "tokenizedCard": {
```

```
    "type": "001"
   },
   "instrumentIdentifier": {
    "id": "7010000000016241111",
    "state": "ACTIVE"
   },
   "paymentInstrument": {
    "id": "080AE120369A7947E063A2598D0A718F"
   },
   "card": {
    "type": "001"
   }
  },
  "pointOfSaleInformation": {
   "terminalId": "111111"
  },
  "processingInformation": {
   "paymentSolution": "015"
  },
  "processorInformation": {
   "paymentAccountReferenceNumber": "V0010013022298169667504231315",
   "approvalCode": "888888",
   "networkTransactionId": "123456789619999",
   "transactionId": "123456789619999",
   "responseCode": "100",
   "avs": {
    "code": "X",
    "codeRaw": "I1"
   }
  },
  "reconciliationId": "62700629BNN13VGW",
  "status": "AUTHORIZED",
  "submitTimeUtc": "2023-10-19T05:02:52Z"
}
```


Example: MIT Delayed Transaction with TMS Customer token

Endpoint:

- Production: POST https://api.cybersource.com/pts/v2/payments
- Test: POST https://apitest.cybersource.com/pts/v2/payments

Request

```
{
  "processingInformation": {
   "authorizationOptions": {
    "initiator": {
     "merchantInitiatedTransaction": {
      "reason": "2"
     }
    }
   }
  },
  "paymentInformation": {
```

```
    "customer": {
     "id": "080AC9AB60C92AA2E063A2598D0A0C74"
    }
   },
   "orderInformation": {
    "amountDetails": {
     "totalAmount": "102.21",
     "currency": "USD"
    }
   }
}
```

Response to a Successful Request

```
{
  "_links": {
    "authReversal": {
      "method": "POST",
      "href": "/pts/v2/payments/6976916433716228003955/reversals"
    },
    "self": {
      "method": "GET",
      "href": "/pts/v2/payments/697691643371622800 3955"
    },
    "capture": {
      "method": "POST",
      "href": "/pts/v2/payments/6976916433716228003955/captures"
    }
  },
  "clientReferenceInformation": {
    "code": "1697691643458"
  },
  "id": "697691643371622800 3955",
  "orderInformation": {
    "amountDetails": {
      "authorizedAmount": "102.21",
      "currency": "USD"
    }
  },
  "paymentAccountInformation": {
    "card": {
      "type": "001"
    }
  },
  "paymentInformation": {
    "tokenizedCard": {
      "type": "001"
    },
    "instrumentIdentifier": {
      "id": "7010000000016241111",
      "state": "ACTIVE"
    },
    "paymentInstrument": {
      "id": "080AE6DB37B09557E063A2598D0AA4C9"
    },
    "card": {
```

```
    "type": "001"
   },
   "customer": {
    "id": "080AC9AB60C92AA2E063A2598D0A0C74"
   }
  },
  "pointOfSaleInformation": {
   "terminalId": "111111"
  },
  "processingInformation": {
   "paymentSolution": "015"
  },
  "processorInformation": {
   "paymentAccountReferenceNumber": "V0010013022298169667504231315",
   "approvalCode": "888888",
   "networkTransactionId": "123456789619999",
   "transactionId": "123456789619999",
   "responseCode": "100",
   "avs": {
    "code": "X",
    "codeRaw": "I1"
   }
  },
  "reconciliationId": "62700435FNN143RY",
  "status": "AUTHORIZED",
  "submitTimeUtc": "2023-10-19T05:00:43Z"
}
```


Merchant-Initiated Incremental Transaction with PAN

An incremental authorization is used to increase the total amount authorized for a payment if the initial authorization does not cover the total cost of goods and services. An incremental transaction is an additional amount to the original authorization. The final authorized total includes amounts for both the initial and the incremental authorizations. To create an incremental transaction using the Business Center, choose one of these options:

- Account Top Up
- No Show

Endpoint

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

Required Fields for Processing Merchant-Initiated Incremental Transactions

This section describes how to process a merchant-initiated incremental transaction using TMS tokens. This procedure expects both customer tokens and payment instrument tokens to already store all the required billing information needed to process a payment.

Use these required fields to process merchant-initiated incremental transactions.

issuerInformation.transactionInformation	**Required only for token transactions with Discover or Diners Club. Set this field to the processorInformation.transactionID field that was in the response message when you obtained the customer's credentials.**
orderInformation.amountDetails.currency	
orderInformation.amountDetails.totalAmount	
processingInformation. authorizationOptions. initiator. merchantInitiatedTransaction. reason	**Set the value to `5`.**
processingInformation. authorizationOptions.initiator. type	**Set the value to `merchant`.**
processorInformation.cardReferenceData	**Required only for token transactions with Discover or Diners Club. Set this field to the processorInformation.cardReferenceData field that was in the response message when you obtained the customer's credentials.**

Card-Specific Required Field for Processing a Merchant-Initiated Transactions

Discover

The listed card requires an additional field:

processingInformation. authorizationOptions. initiator. merchantInitiatedTransaction. originalAuthorizedAmount	**Provide the original transaction amount.**

REST Example: Processing Merchant-Initiated Incremental Transactions

Endpoint:

- Send a `PATCH` request to this endpoint: `POST https://api.cybersource.com/pts/v2/payments`

{id} is the original authorization ID.

Request

```
{
   "orderInformation": {
  "billTo" : {
     "country" : "US",
     "lastName" : "Kim",
     "address1" : "201 S. Division St.",
     "postalCode" : "48104-2201",
     "locality" : "Ann Arbor",
     "administrativeArea" : "MI",
     "firstName" : "Kyong-Jin",
         "phoneNumber": "5554327113",
     "email" : "test@cybs.com"
    },
      "amountDetails": {
         "totalAmount": "120.00",
         "currency": "USD"
      }
   },
   "paymentInformation": {
      "card": {
         "expirationYear": "2031",
         "number": "4111xxxxxxxxxxx",
         "expirationMonth": "12"
      }
   },
   "processingInformation": {
      "authorizationOptions": {
         "initiator": {
       "type": "merchant",
          "merchantInitiatedTransaction": {
           "originalAuthorizedAmount": "100",
                    // Required for Discover
           "previousTransactionId": "123456789619999",
           "reason": "5"
          }
         }
      }
   }
}
```

Response to a Successful Request

```
{
   "_links": {
      "authReversal": {
         "method": "POST",
         "href": "/pts/v2/payments/6533225006556860003002/reversals"
      },
      "self": {
         "method": "GET",
         "href": "/pts/v2/payments/6533225006556860003002"
      },
```

```
      "capture": {
        "method": "POST",
        "href": "/pts/v2/payments/6533225006556860003002/captures"
      }
    },
    "clientReferenceInformation": {
      "code": "1653322500637"
    },
    "id": "6533225006556860003002",
    "orderInformation": {
      "amountDetails": {
        "authorizedAmount": "100.00",
        "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": "64143477A3AJ4P2Z",
    "status": "AUTHORIZED",
    "submitTimeUtc": "2022-05-23T16:15:00Z"
}
```


Merchant-Initiated Incremental Transaction with TMS

An incremental authorization is used to increase the total amount authorized for a payment if the initial authorization does not cover the total cost of goods and services.

An incremental transaction is an additional amount to the original authorization. The final authorized total includes amounts for both the initial and the incremental authorizations. Incremental transactions are limited to certain merchant categories, such as rental, lodging, transit, amusement parks, restaurants, and bars.
This section describes how to process a merchant-initiated incremental transaction using these TMS token types:

Customer	Customer tokens store one or more customer payment instrument tokens and shipping address tokens. Including a customer token eliminates the need to include billing information, card information, and the previous transaction's ID. `"paymentInformation": { "customer": { "id": "07C9CA98022DA498E063A2598D0AA400" } }` For more information about this TMS token type, see *Customer Tokens* in the Token Management Service Developer Guide.
Payment Instrument	Payment instrument tokens store an instrument identifier token, card information, and billing information. Payment instruments are not linked to a customer token. Including a payment instrument eliminates the need to include billing information, card information, and the previous transaction's ID. `"paymentInformation": { "paymentInstrument": { "id": "07CA24EF20F9E2C9E063A2598D0A8565" } }` For more information about this TMS token type, see *Payment Instrument Token* in the Token Management Service Developer Guide.
Instrument Identifier	Instrument identifier tokens store only a PAN. Including an instrument identifier

eliminates the need to include a PAN and the previous transaction's ID.

```
"paymentInformation": {
 "instrumentIdentifier": {
  "id": "7010000000016241111"
 }
}
```

For more information about this TMS token type, see *Instrument Identifier Token* in the Token Management Service Developer Guide.

To create an incremental transaction using the Business Center, choose one of these options:

- Account Top Up
- No Show

Endpoint

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

Required Fields for MIT Incremental Transaction with TMS

Include these Required Fields

orderInformation.amountDetails.currency

orderInformation.amountDetails.totalAmount

paymentInformation.[tokentype].id

Where **[tokentype]** is the TMS token type you are using:

- **customer**
- **instrumentIdentifier**
- **paymentInstrument**

processingInformation. authorizationOptions.initiator. merchantInitiatedTransaction. reason

Set the value to `5`.

Instrument Identifier Required Fields

If you are using the **paymentInformation.instrumentIdentifier.id** token, include these required fields in addition to the required fields listed above.

orderInformation.billTo.address1

orderInformation.billTo.administrativeArea

orderInformation.billTo.country

orderInformation.billTo.email

orderInformation.billTo.firstName

orderInformation.billTo.lastName

orderInformation.billTo.locality

orderInformation.billTo.phoneNumber

orderInformation.billTo.postalCode

paymentInformation.card.expirationMonth

paymentInformation.card.expirationYear

Card-Specific Fields

Include these fields when processing an authorization with these card types. The listed card type requires an additional field.

Diners Club	**processorInformation.cardReferenceData:** **Required only for token transactions. Set this field to the processorInformation.cardReferenceData field that was in the response message when you obtained the customer's credentials.** **issuerInformation.transactionInformation:** **Required only for token transactions. Set this field to the processorInformation.transactionID field that was in the response message when you obtained the customer's credentials.**
Discover	**processingInformation.authorizationOptions.initiator.merchantInitiatedTransaction.originalAuthorizedAmou** **Set to the original transaction amount.** **processorInformation.cardReferenceData** **Required only for token transactions. Set this field to the processorInformation.cardReferenceData field that was in the response message when you obtained the customer's credentials.** **issuerInformation.transactionInformation** **Required only for token transactions. Set this field to the processorInformation.transactionID field**

that was in the response message when you obtained the customer's credentials.

Example: MIT Incremental Transaction with a TMS Instrument Identifier

Endpoint:

- Production: POST https://api.cybersource.com/pts/v2/payments
- Test: POST https://apitest.cybersource.com/pts/v2/payments

Request

```
{
 "processingInformation": {
  "authorizationOptions": {
   "initiator": {
    "merchantInitiatedTransaction": {
     "reason": "5"
    }
   }
  }
 },
 "paymentInformation": {
  "card": {
   "expirationMonth": "12",
   "expirationYear": "2031"
  },
  "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"
  }
 }
}
```

Response to a Successful Request

```
{
```

```
  "_links": {
    "authReversal": {
      "method": "POST",
      "href": "/pts/v2/payments/6976922830456934003954/reversals"
    },
    "self": {
      "method": "GET",
      "href": "/pts/v2/payments/697692283045693400 3954"
    },
    "capture": {
      "method": "POST",
      "href": "/pts/v2/payments/6976922830456934003954/captures"
    }
  },
  "clientReferenceInformation": {
    "code": "1697692283160"
  },
  "id": "6976922830456934003954",
  "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"
  },
  "processingInformation": {
    "paymentSolution": "015"
  },
  "processorInformation": {
    "paymentAccountReferenceNumber": "V0010013022298169667504231315",
    "approvalCode": "888888",
    "networkTransactionId": "123456789619999",
    "transactionId": "123456789619999",
    "responseCode": "100",
    "avs": {
      "code": "X",
      "codeRaw": "I1"
```

```
   }
  },
  "reconciliationId": "62700184NNMR6XFK",
  "status": "AUTHORIZED",
  "submitTimeUtc": "2023-10-19T05:11:23Z"
}
```

Example: MIT Incremental Transaction with a TMS Payment Instrument

Endpoint:

- Production: POST https://api.cybersource.com/pts/v2/payments
- Test: POST https://apitest.cybersource.com/pts/v2/payments

Request

```
{
  "processingInformation": {
    "authorizationOptions": {
      "initiator": {
        "merchantInitiatedTransaction": {
          "reason": "5"
        }
      }
    }
  },
  "paymentInformation": {
    "paymentInstrument": {
      "id": "080AE120369A7947E063A2598D0A718F"
    }
  },
  "orderInformation": {
    "amountDetails": {
      "totalAmount": "102.21",
      "currency": "USD"
    }
  }
}
```

Response to a Successful Request

```
{
  "_links": {
    "authReversal": {
      "method": "POST",
      "href": "/pts/v2/payments/6976917718796256603955/reversals"
    },
    "self": {
      "method": "GET",
      "href": "/pts/v2/payments/697691771879625660 3955"
    },
    "capture": {
      "method": "POST",
      "href": "/pts/v2/payments/6976917718796256603955/captures"
```

```
    }
  },
  "clientReferenceInformation": {
    "code": "1697691771976"
  },
  "id": "6976917718796256603955",
  "orderInformation": {
    "amountDetails": {
      "authorizedAmount": "102.21",
      "currency": "USD"
    }
  },
  "paymentAccountInformation": {
    "card": {
      "type": "001"
    }
  },
  "paymentInformation": {
    "tokenizedCard": {
      "type": "001"
    },
    "instrumentIdentifier": {
      "id": "7010000000016241111",
      "state": "ACTIVE"
    },
    "paymentInstrument": {
      "id": "080AE120369A7947E063A2598D0A718F"
    },
    "card": {
      "type": "001"
    }
  },
  "pointOfSaleInformation": {
    "terminalId": "111111"
  },
  "processingInformation": {
    "paymentSolution": "015"
  },
  "processorInformation": {
    "paymentAccountReferenceNumber": "V0010013022298169667504231315",
    "approvalCode": "888888",
    "networkTransactionId": "123456789619999",
    "transactionId": "123456789619999",
    "responseCode": "100",
    "avs": {
      "code": "X",
      "codeRaw": "I1"
    }
  },
  "reconciliationId": "62700629BNN13VGW",
  "status": "AUTHORIZED",
  "submitTimeUtc": "2023-10-19T05:02:52Z"
}
```

Example: MIT Incremental Transaction with a TMS Customer token

Endpoint:

- Production: POST https://api.cybersource.com/pts/v2/payments
- Test: POST https://apitest.cybersource.com/pts/v2/payments

Request

```
{
 "processingInformation": {
  "authorizationOptions": {
   "initiator": {
    "merchantInitiatedTransaction": {
     "reason": "5"
    }
   }
  }
 },
 "paymentInformation": {
  "customer": {
   "id": "080AC9AB60C92AA2E063A2598D0A0C74"
  }
 },
 "orderInformation": {
  "amountDetails": {
   "totalAmount": "102.21",
   "currency": "USD"
  }
 }
}
```

Response to a Successful Request

```
{
 "_links": {
  "authReversal": {
   "method": "POST",
   "href": "/pts/v2/payments/6976916433716228003955/reversals"
  },
  "self": {
   "method": "GET",
   "href": "/pts/v2/payments/697691643371622800 3955"
  },
  "capture": {
   "method": "POST",
   "href": "/pts/v2/payments/6976916433716228003955/captures"
  }
 },
 "clientReferenceInformation": {
  "code": "1697691643458"
 },
 "id": "6976916433716228003955",
 "orderInformation": {
  "amountDetails": {
   "authorizedAmount": "102.21",
```

```
    "currency": "USD"
   }
  },
  "paymentAccountInformation": {
   "card": {
    "type": "001"
   }
  },
  "paymentInformation": {
   "tokenizedCard": {
    "type": "001"
   },
   "instrumentIdentifier": {
    "id": "7010000000016241111",
    "state": "ACTIVE"
   },
   "paymentInstrument": {
    "id": "080AE6DB37B09557E063A2598D0AA4C9"
   },
   "card": {
    "type": "001"
   },
   "customer": {
    "id": "080AC9AB60C92AA2E063A2598D0A0C74"
   }
  },
  "pointOfSaleInformation": {
   "terminalId": "111111"
  },
  "processingInformation": {
   "paymentSolution": "015"
  },
  "processorInformation": {
   "paymentAccountReferenceNumber": "V0010013022298169667504231315",
   "approvalCode": "888888",
   "networkTransactionId": "123456789619999",
   "transactionId": "123456789619999",
   "responseCode": "100",
   "avs": {
    "code": "X",
    "codeRaw": "I1"
   }
  },
  "reconciliationId": "62700435FNN143RY",
  "status": "AUTHORIZED",
  "submitTimeUtc": "2023-10-19T05:00:43Z"
}
```

Merchant-Initiated No-Show Transactions with PAN

A no-show authorization occurs when a merchant charges a customer after the customer makes a reservation, and does not show up to claim the reservation. In this situation, the customer is charged an agreed upon fee for not showing up as expected.

Endpoint

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

Required Fields for Processing Merchant-Initiated No-Show Charges

Use these required fields to process a merchant-initiated no-show charges transaction.

issuerInformation.transactionInformation	**Required only for token transactions with Discover or Diners Club. Set this field to the processorInformation.transactionID field that was in the response message when you obtained the customer's credentials.**
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.phoneNumber	
orderInformation.billTo.postalCode	
paymentInformation.card.expirationMonth	
paymentInformation.card.expirationYear	
paymentInformation.card.number	
processingInformation. authorizationOptions. initiator.	**For Discover and American Express cards, use the transaction ID from the**

merchantInitiatedTransaction. previousTransactionId	**original transaction. For Visa, use the last successful transaction ID.**
processingInformation. authorizationOptions. initiator. merchantInitiatedTransaction. reason	**Set the value to `4`.**
processingInformation. authorizationOptions. initiator. type	**Set the value to `merchant`.**
processorInformation.cardReferenceData	**Required only for token transactions with Discover or Diners Club. Set this field to the processorInformation.cardReferenceData field that was in the response message when you obtained the customer's credentials.**

Card-Specific Required Field for Processing a Merchant-Initiated Transactions

Discover

The listed card requires an additional field:

processingInformation. authorizationOptions. initiator. merchantInitiatedTransaction. originalAuthorizedAmount	**Provide the original transaction amount.**

Optional Field for Processing Merchant-Initiated No-Show Charges

You can use these optional fields to include additional information when authorizing a request for an MIT no-show charge:

processingInformation. authorizationOptions. initiator. storedCredentialUsed	**If the payment information is COF information, set the value to `true`.**

REST Example: Processing Merchant-Initiated No-Show Transactions

Endpoint:

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

Request

```
{
  "processingInformation": {
    "authorizationOptions": {
```

```
        "initiator": {
       "type": "merchant",
        "merchantInitiatedTransaction": {
         "originalAuthorizedAmount": "100", //Discover only
         "previousTransactionId": "123456789619999",
         "reason": "4"
        }
       }
      }
   },
   "orderInformation": {
  "billTo" : {
    "country" : "US",
    "lastName" : "Kim",
    "address1" : "201 S. Division St.",
    "postalCode" : "48104-2201",
    "locality" : "Ann Arbor",
    "administrativeArea" : "MI",
    "firstName" : "Kyong-Jin",
        "phoneNumber": "5554327113",
    "email" : "test@cybs.com"
   },
      "amountDetails": {
        "totalAmount": "150.00",
        "currency": "USD"
      }
   },
   "paymentInformation": {
      "card": {
        "expirationYear": "2031",
        "number": "4111xxxxxxxxxxx",
        "expirationMonth": "12"
      }
   }
}
```

Response to a Successful Request

```
{
   "_links": {
      "authReversal": {
        "method": "POST",
        "href": "/pts/v2/payments/6534214295466223903006/reversals"
      },
      "self": {
        "method": "GET",
        "href": "/pts/v2/payments/653421429546622390306"
      },
      "capture": {
        "method": "POST",
        "href": "/pts/v2/payments/6534214295466223903006/captures"
      }
   },
   "clientReferenceInformation": {
      "code": "1653421429522"
   },
```

```
  "id": "6534214295466223903006",
  "orderInformation": {
    "amountDetails": {
      "authorizedAmount": "150.00",
      "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": "64365823G3K7HFAM",
  "status": "AUTHORIZED",
  "submitTimeUtc": "2022-05-24T19:43:49Z"
}
```


Merchant-Initiated No-Show Transaction with TMS

A no-show authorization occurs when a merchant charges a customer after the customer makes a reservation, and does not show up to claim the reservation. In this situation, the customer is charged an agreed upon fee for not showing up as expected.
This section describes how to process a merchant-initiated no-show transaction using these TMS token types:

Customer	Customer tokens store one or more customer payment instrument tokens and shipping address tokens.

	Including a customer token eliminates the need to include billing information, card information, and the previous transaction's ID.
	
	`"paymentInformation": {`
	`"customer": {`
	`"id": "07C9CA98022DA498E063A2598D0AA400"`
	`}`
	`}`
	
	For more information about this TMS token type, see *Customer Tokens* in the Token Management Service Developer Guide.

Payment Instrument	Payment instrument tokens store an instrument identifier token, card information, and billing information. Payment instruments are not linked to a customer token.
	
	Including a payment instrument eliminates the need to include billing information, card information, and the previous transaction's ID.
	
	`"paymentInformation": {`
	`"paymentInstrument": {`
	`"id": "07CA24EF20F9E2C9E063A2598D0A8565"`
	`}`
	`}`
	
	For more information about this TMS token type, see *Payment Instrument Token* in the Token Management Service Developer Guide.

Instrument Identifier	Instrument identifier tokens store only a PAN. Including an instrument identifier eliminates the need to include a PAN and the previous transaction's ID.
	
	`"paymentInformation": {`
	`"instrumentIdentifier": {`
	`"id": "7010000000016241111"`
	`}`
	`}`
	
	For more information about this TMS token type, see *Instrument Identifier Token* in the Token Management Service Developer Guide.

Endpoint

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

Required Fields for MIT No-Show Transaction with TMS

Include these Required Fields

orderInformation.amountDetails.currency

orderInformation.amountDetails.totalAmount

paymentInformation.[tokentype].id	Where **[tokentype]** is the TMS token type you are using: • **customer** • **instrumentIdentifier** • **paymentInstrument**
processingInformation. authorizationOptions. initiator. merchantInitiatedTransaction.reason	Set the value to 4.

Instrument Identifier Required Fields

If you are using the **paymentInformation.instrumentIdentifier.id** token, include these required fields in addition to the required fields listed above.

orderInformation.billTo.address1

orderInformation.billTo.administrativeArea

orderInformation.billTo.country

orderInformation.billTo.email

orderInformation.billTo.firstName

orderInformation.billTo.lastName

orderInformation.billTo.locality

orderInformation.billTo.phoneNumber

orderInformation.billTo.postalCode

paymentInformation.card.expirationMonth

paymentInformation.card.expirationYear

Card-Specific Fields

Include these fields when processing an authorization with these card types.
The listed card type requires an additional field.

Diners Club	**processorInformation.cardReferenceData:** **Required only for token transactions. Set this field to the processorInformation.cardReferenceDatafield that was in the response message when you obtained the customer's credentials.** **issuerInformation.transactionInformation:** **Required only for token transactions. Set this field to the processorInformation.transactionIDfield that was in the response message when you obtained the customer's credentials.**
Discover	**processingInformation.authorizationOptions.initiator.merchantInitiatedTransaction.originalAuthorizedAmou** **Set to the original transaction amount.** **processorInformation.cardReferenceData** **Required only for token transactions. Set this field to the processorInformation.cardReferenceDatafield that was in the response message when you obtained the customer's credentials.** **issuerInformation.transactionInformation** **Required only for token transactions. Set this field to the processorInformation.transactionIDfield that was in the response message when you obtained the customer's credentials.**

Example: MIT No-Show Transaction with a TMS Instrument Identifier

Endpoint:

- Production: POST https://api.cybersource.com/pts/v2/payments
- Test: POST https://apitest.cybersource.com/pts/v2/payments

Request

```
{
  "processingInformation": {
    "authorizationOptions": {
      "initiator": {
        "merchantInitiatedTransaction": {
          "reason": "4"
        }
      }
    }
  },
```

```
  "paymentInformation": {
   "card": {
    "expirationMonth": "12",
    "expirationYear": "2031"
   },
   "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"
   }
  }
}
```

Response to a Successful Request

```
{
  "_links": {
   "authReversal": {
    "method": "POST",
    "href": "/pts/v2/payments/6976922830456934003954/reversals"
   },
   "self": {
    "method": "GET",
    "href": "/pts/v2/payments/697692283045693400 3954"
   },
   "capture": {
    "method": "POST",
    "href": "/pts/v2/payments/6976922830456934003954/captures"
   }
  },
  "clientReferenceInformation": {
   "code": "1697692283160"
  },
  "id": "6976922830456934003954",
  "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"
   },
   "processingInformation": {
    "paymentSolution": "015"
   },
   "processorInformation": {
    "paymentAccountReferenceNumber": "V0010013022298169667504231315",
    "approvalCode": "888888",
    "networkTransactionId": "123456789619999",
    "transactionId": "123456789619999",
    "responseCode": "100",
    "avs": {
     "code": "X",
     "codeRaw": "I1"
    }
   },
   "reconciliationId": "62700184NNMR6XFK",
   "status": "AUTHORIZED",
   "submitTimeUtc": "2023-10-19T05:11:23Z"
}
```


Example: MIT No-Show Transaction with a TMS Payment Instrument

Endpoint:

- Production: POST https://api.cybersource.com/pts/v2/payments
- Test: POST https://apitest.cybersource.com/pts/v2/payments

Request


```
{
  "processingInformation": {
    "authorizationOptions": {
      "initiator": {
        "merchantInitiatedTransaction": {
          "reason": "4"
        }
      }
```

```
    }
   },
   "paymentInformation": {
    "paymentInstrument": {
     "id": "080AE120369A7947E063A2598D0A718F"
    }
   },
   "orderInformation": {
    "amountDetails": {
     "totalAmount": "102.21",
     "currency": "USD"
    }
   }
}
```

Response to a Successful Request

```
{
  "_links": {
    "authReversal": {
      "method": "POST",
      "href": "/pts/v2/payments/6976917718796256603955/reversals"
    },
    "self": {
      "method": "GET",
      "href": "/pts/v2/payments/697691771879625660 3955"
    },
    "capture": {
      "method": "POST",
      "href": "/pts/v2/payments/6976917718796256603955/captures"
    }
  },
  "clientReferenceInformation": {
    "code": "1697691771976"
  },
  "id": "6976917718796256603955",
  "orderInformation": {
    "amountDetails": {
      "authorizedAmount": "102.21",
      "currency": "USD"
    }
  },
  "paymentAccountInformation": {
    "card": {
      "type": "001"
    }
  },
  "paymentInformation": {
    "tokenizedCard": {
      "type": "001"
    },
    "instrumentIdentifier": {
      "id": "7010000000016241111",
      "state": "ACTIVE"
    },
    "paymentInstrument": {
```

```
    "id": "080AE120369A7947E063A2598D0A718F"
   },
   "card": {
    "type": "001"
   }
  },
  "pointOfSaleInformation": {
   "terminalId": "111111"
  },
  "processingInformation": {
   "paymentSolution": "015"
  },
  "processorInformation": {
   "paymentAccountReferenceNumber": "V001001302229816966750 4231315",
   "approvalCode": "888888",
   "networkTransactionId": "123456789619999",
   "transactionId": "123456789619999",
   "responseCode": "100",
   "avs": {
    "code": "X",
    "codeRaw": "I1"
   }
  },
  "reconciliationId": "62700629BNN13VGW",
  "status": "AUTHORIZED",
  "submitTimeUtc": "2023-10-19T05:02:52Z"
}
```


Example: MIT No-Show Transaction with a TMS Customer

Endpoint:

- Production: POST https://api.cybersource.com/pts/v2/payments
- Test: POST https://apitest.cybersource.com/pts/v2/payments

Request


```
{
  "processingInformation": {
   "authorizationOptions": {
    "initiator": {
     "merchantInitiatedTransaction": {
      "reason": "4"
     }
    }
   }
  },
  "paymentInformation": {
   "customer": {
    "id": "080AC9AB60C92AA2E063A2598D0A0C74"
   }
  },
  "orderInformation": {
   "amountDetails": {
    "totalAmount": "102.21",
```

```
    "currency": "USD"
   }
  }
}
```

Response to a Successful Request

```
{
 "_links": {
  "authReversal": {
   "method": "POST",
   "href": "/pts/v2/payments/6976916433716228003955/reversals"
  },
  "self": {
   "method": "GET",
   "href": "/pts/v2/payments/697691643371622800 3955"
  },
  "capture": {
   "method": "POST",
   "href": "/pts/v2/payments/6976916433716228003955/captures"
  }
 },
 "clientReferenceInformation": {
  "code": "1697691643458"
 },
 "id": "6976916433716228003955",
 "orderInformation": {
  "amountDetails": {
   "authorizedAmount": "102.21",
   "currency": "USD"
  }
 },
 "paymentAccountInformation": {
  "card": {
   "type": "001"
  }
 },
 "paymentInformation": {
  "tokenizedCard": {
   "type": "001"
  },
  "instrumentIdentifier": {
   "id": "7010000000016241111",
   "state": "ACTIVE"
  },
  "paymentInstrument": {
   "id": "080AE6DB37B09557E063A2598D0AA4C9"
  },
  "card": {
   "type": "001"
  },
  "customer": {
   "id": "080AC9AB60C92AA2E063A2598D0A0C74"
  }
 },
 "pointOfSaleInformation": {
```

```
    "terminalId": "111111"
  },
  "processingInformation": {
    "paymentSolution": "015"
  },
  "processorInformation": {
    "paymentAccountReferenceNumber": "V0010013022298169667504231315",
    "approvalCode": "888888",
    "networkTransactionId": "123456789619999",
    "transactionId": "123456789619999",
    "responseCode": "100",
    "avs": {
      "code": "X",
      "codeRaw": "I1"
    }
  },
  "reconciliationId": "62700435FNN143RY",
  "status": "AUTHORIZED",
  "submitTimeUtc": "2023-10-19T05:00:43Z"
}
```


Merchant-Initiated Reauthorization Transactions with PAN

A reauthorization occurs when the completion or fulfillment of the original order or service extends beyond the authorized amount time limit. There are two common reauthorization scenarios:

- Split or delayed shipments by a retailer
- Extended car rentals, hotel stays, or cruise line bookings

Endpoint

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

Required Fields for Processing Merchant-Initiated Reauthorized Transactions

Use these required fields to process a merchant-initiated reauthorization transaction.

> **Important**
>
> When using relaxed requirements for address data and the expiration date, not all fields in this list are required. It is your responsibility to determine whether your account is enabled to use this feature and which fields are required. For details

about relaxed requirements, see *Relaxed Requirements for Address Data and Expiration Date in Payment Transactions* on page 264.

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.phoneNumber

orderInformation.billTo.postalCode

paymentInformation.card.expirationMonth

paymentInformation.card.expirationYear

paymentInformation.card.number

processingInformation. authorizationOptions. initiator. merchantInitiatedTransaction. previousTransactionId	For Discover and American Express cards, use the transaction ID from the original transaction. For Visa, use the last successful transaction ID.
processingInformation. authorizationOptions. initiator. merchantInitiatedTransaction. reason	**Set the value to** 3**.**
processingInformation. authorizationOptions. initiator. type	**Set the value to** merchant**.**

Card-Specific Required Field for Processing a Merchant-Initiated Transactions

Discover

The listed card requires an additional field:

processingInformation. authorizationOptions. initiator. merchantInitiatedTransaction. originalAuthorizedAmount	**Provide the original transaction amount.**

REST Example: Processing a Merchant-Initiated Reauthorized Transaction

Endpoint:

- Production: POST https://api.cybersource.com/pts/v2/payments
- Test: POST https://apitest.cybersource.com/pts/v2/payments

Request

```
{
   "processingInformation": {
      "authorizationOptions": {
         "initiator": {
       "type": "merchant",
          "merchantInitiatedTransaction": {
           "originalAuthorizedAmount": "100", // Discover Only
           "previousTransactionId": "123456789619999",
           "reason": "3"
          }
         }
      }
   },
   "orderInformation": {
  "billTo" : {
     "country" : "US",
     "lastName" : "Kim",
     "address1" : "201 S. Division St.",
     "postalCode" : "48104-2201",
     "locality" : "Ann Arbor",
     "administrativeArea" : "MI",
     "firstName" : "Kyong-Jin",
         "phoneNumber": "5554327113",
     "email" : "test@cybs.com"
    },
      "amountDetails": {
         "totalAmount": "130.00",
         "currency": "USD"
      }
   },
   "paymentInformation": {
      "card": {
         "expirationYear": "2031",
         "number": "4111xxxxxxxxxxx",
         "expirationMonth": "12"
      }
   }
}
```

Response to a Successful Request

```
{
   "_links": {
      "authReversal": {
         "method": "POST",
```

```
        "href": "/pts/v2/payments/6541178668686490403003/reversals"
      },
      "self": {
        "method": "GET",
        "href": "/pts/v2/payments/654117866868649040300 3"
      },
      "capture": {
        "method": "POST",
        "href": "/pts/v2/payments/6541178668686490403003/captures"
      }
    },
    "clientReferenceInformation": {
      "code": "1654117866849"
    },
    "id": "6541178668686490403003",
    "orderInformation": {
      "amountDetails": {
        "authorizedAmount": "130.00",
        "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": "65313868D3TXXC05",
    "status": "AUTHORIZED",
    "submitTimeUtc": "2022-06-01T21:11:06Z"
}
```

Merchant-Initiated Reauthorization Transactions with TMS

A reauthorization occurs when the completion or fulfillment of the original order or service extends beyond the authorized amount time limit. There are two common reauthorization scenarios:

- Split or delayed shipments by a retailer
- Extended car rentals, hotel stays, or cruise line bookings

This section describes how to process a merchant-initiated reauthorization transactions using one or more TMS token types:

Customer	Customer tokens store one or more customer payment instrument tokens and shipping address tokens. Including a customer token eliminates the need to include billing information, card information, and the previous transaction's ID. `"paymentInformation": { "customer": { "id": "07C9CA98022DA498E063A2598D0AA400" } }` For more information about this TMS token type, see *Customer Tokens* in the Token Management Service Developer Guide.
Payment Instrument	Payment instrument tokens store an instrument identifier token, card information, and billing information. Payment instruments are not linked to a customer token. Including a payment instrument eliminates the need to include billing information, card information, and the previous transaction's ID. `"paymentInformation": { "paymentInstrument": { "id": "07CA24EF20F9E2C9E063A2598D0A8565" } }`

	For more information about this TMS token type, see *Payment Instrument Token* in the Token Management Service Developer Guide.
Instrument Identifier	Instrument identifier tokens store only a PAN. Including an instrument identifier eliminates the need to include a PAN and the previous transaction's ID. `"paymentInformation": { "instrumentIdentifier": { "id": "7010000000016241111" } }` For more information about this TMS token type, see *Instrument Identifier Token* in the Token Management Service Developer Guide.

Endpoint

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

Required Fields for MIT Reauthorization Transaction with TMS

Include these Required Fields

orderInformation.amountDetails.currency

orderInformation.amountDetails.totalAmount

paymentInformation.[tokentype].id	Where **[tokentype]** is the TMS token type you are using: • **customer** • **instrumentIdentifier** • **paymentInstrument**
processingInformation. authorizationOptions. initiator. merchantInitiatedTransaction. reason	Set the value to 3.

Instrument Identifier Required Fields

If you are using the **paymentInformation.instrumentIdentifier.id** token, include these required fields in addition to the required fields listed above.

orderInformation.billTo.address1

orderInformation.billTo.administrativeArea

orderInformation.billTo.country

orderInformation.billTo.email

orderInformation.billTo.firstName

orderInformation.billTo.lastName

orderInformation.billTo.locality

orderInformation.billTo.phoneNumber

orderInformation.billTo.postalCode

paymentInformation.card.expirationMonth

paymentInformation.card.expirationYear

Card-Specific Fields

Include these fields when processing an authorization with these card types. The listed card type requires an additional field.

Diners Club	**processorInformation.cardReferenceData:** **Required only for token transactions. Set this field to the processorInformation.cardReferenceData field that was in the response message when you obtained the customer's credentials.** **issuerInformation.transactionInformation:** **Required only for token transactions. Set this field to the processorInformation.transactionID field that was in the response message when you obtained the customer's credentials.**
Discover	**processingInformation.authorizationOptions.initiator.merchantInitiatedTransaction.originalAuthorizedAmou** **Set to the original transaction amount.** **processorInformation.cardReferenceData** **Required only for token transactions. Set this field to the processorInformation.cardReferenceData field that was in the response message when you obtained the customer's credentials.** **issuerInformation.transactionInformation**

	Required only for token transactions. Set this field to the processorInformation.transactionID field that was in the response message when you obtained the customer's credentials.

Example: MIT Reauthorization Transaction with a TMS Instrument Identifier

Endpoint:

- Production: POST https://api.cybersource.com/pts/v2/payments
- Test: POST https://apitest.cybersource.com/pts/v2/payments

Request

```
{
 "processingInformation": {
  "authorizationOptions": {
   "initiator": {
    "merchantInitiatedTransaction": {
     "reason": "3"
    }
   }
  }
 },
 "paymentInformation": {
  "card": {
   "expirationMonth": "12",
   "expirationYear": "2031"
  },
  "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"
  }
 }
}
```

Response to a Successful Request


```
{
 "_links": {
  "authReversal": {
   "method": "POST",
   "href": "/pts/v2/payments/697692283045693400 3954/reversals"
  },
  "self": {
   "method": "GET",
   "href": "/pts/v2/payments/697692283045693400 3954"
  },
  "capture": {
   "method": "POST",
   "href": "/pts/v2/payments/697692283045693400 3954/captures"
  }
 },
 "clientReferenceInformation": {
  "code": "1697692283160"
 },
 "id": "6976922830456934003954",
 "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"
 },
 "processingInformation": {
  "paymentSolution": "015"
 },
 "processorInformation": {
  "paymentAccountReferenceNumber": "V0010013022298169667504231315",
  "approvalCode": "888888",
  "networkTransactionId": "123456789619999",
  "transactionId": "123456789619999",
  "responseCode": "100",
```

```
    "avs": {
     "code": "X",
     "codeRaw": "I1"
    }
   },
   "reconciliationId": "62700184NNMR6XFK",
   "status": "AUTHORIZED",
   "submitTimeUtc": "2023-10-19T05:11:23Z"
}
```


Example: MIT Reauthorization Transaction with a TMS Payment Instrument

Endpoint:

- Production: POST https://api.cybersource.com/pts/v2/payments
- Test: POST https://apitest.cybersource.com/pts/v2/payments

Request

```
{
 "processingInformation": {
  "authorizationOptions": {
   "initiator": {
    "merchantInitiatedTransaction": {
     "reason": "3"
    }
   }
  }
 },
 "paymentInformation": {
  "paymentInstrument": {
   "id": "080AE120369A7947E063A2598D0A718F"
  }
 },
 "orderInformation": {
  "amountDetails": {
   "totalAmount": "102.21",
   "currency": "USD"
  }
 }
}
```

Response to a Successful Request


```
{
 "_links": {
  "authReversal": {
   "method": "POST",
   "href": "/pts/v2/payments/6976917718796256603955/reversals"
  },
  "self": {
   "method": "GET",
   "href": "/pts/v2/payments/6976917718796256603955"
  },
```

```
    "capture": {
      "method": "POST",
      "href": "/pts/v2/payments/6976917718796256603955/captures"
    }
  },
  "clientReferenceInformation": {
    "code": "1697691771976"
  },
  "id": "6976917718796256603955",
  "orderInformation": {
    "amountDetails": {
      "authorizedAmount": "102.21",
      "currency": "USD"
    }
  },
  "paymentAccountInformation": {
    "card": {
      "type": "001"
    }
  },
  "paymentInformation": {
    "tokenizedCard": {
      "type": "001"
    },
    "instrumentIdentifier": {
      "id": "7010000000016241111",
      "state": "ACTIVE"
    },
    "paymentInstrument": {
      "id": "080AE120369A7947E063A2598D0A718F"
    },
    "card": {
      "type": "001"
    }
  },
  "pointOfSaleInformation": {
    "terminalId": "111111"
  },
  "processingInformation": {
    "paymentSolution": "015"
  },
  "processorInformation": {
    "paymentAccountReferenceNumber": "V0010013022298169667504231315",
    "approvalCode": "888888",
    "networkTransactionId": "123456789619999",
    "transactionId": "123456789619999",
    "responseCode": "100",
    "avs": {
      "code": "X",
      "codeRaw": "I1"
    }
  },
  "reconciliationId": "62700629BNN13VGW",
  "status": "AUTHORIZED",
  "submitTimeUtc": "2023-10-19T05:02:52Z"
}
```

Example: MIT Reauthorization Transaction with a TMS Customer

Endpoint:

- Production: POST https://api.cybersource.com/pts/v2/payments
- Test: POST https://apitest.cybersource.com/pts/v2/payments

Request

```
{
 "processingInformation": {
  "authorizationOptions": {
   "initiator": {
    "merchantInitiatedTransaction": {
     "reason": "3"
    }
   }
  }
 },
 "paymentInformation": {
  "customer": {
   "id": "080AC9AB60C92AA2E063A2598D0A0C74"
  }
 },
 "orderInformation": {
  "amountDetails": {
   "totalAmount": "102.21",
   "currency": "USD"
  }
 }
}
```

Response to a Successful Request

```
{
 "_links": {
  "authReversal": {
   "method": "POST",
   "href": "/pts/v2/payments/6976916433716228003955/reversals"
  },
  "self": {
   "method": "GET",
   "href": "/pts/v2/payments/697691643371622800 3955"
  },
  "capture": {
   "method": "POST",
   "href": "/pts/v2/payments/6976916433716228003955/captures"
  }
 },
 "clientReferenceInformation": {
  "code": "1697691643458"
 },
 "id": "6976916433716228003955",
 "orderInformation": {
  "amountDetails": {
   "authorizedAmount": "102.21",
```

```
    "currency": "USD"
   }
  },
  "paymentAccountInformation": {
   "card": {
    "type": "001"
   }
  },
  "paymentInformation": {
   "tokenizedCard": {
    "type": "001"
   },
   "instrumentIdentifier": {
    "id": "7010000000016241111",
    "state": "ACTIVE"
   },
   "paymentInstrument": {
    "id": "080AE6DB37B09557E063A2598D0AA4C9"
   },
   "card": {
    "type": "001"
   },
   "customer": {
    "id": "080AC9AB60C92AA2E063A2598D0A0C74"
   }
  },
  "pointOfSaleInformation": {
   "terminalId": "111111"
  },
  "processingInformation": {
   "paymentSolution": "015"
  },
  "processorInformation": {
   "paymentAccountReferenceNumber": "V0010013022298169667504231315",
   "approvalCode": "888888",
   "networkTransactionId": "123456789619999",
   "transactionId": "123456789619999",
   "responseCode": "100",
   "avs": {
    "code": "X",
    "codeRaw": "I1"
   }
  },
  "reconciliationId": "62700435FNN143RY",
  "status": "AUTHORIZED",
  "submitTimeUtc": "2023-10-19T05:00:43Z"
}
```

Merchant-Initiated Resubmission Transaction with PAN

A resubmission transaction is used when a merchant resubmits an authorization to recover an outstanding debt from the customer. A common scenario is when a card was initially declined due to insufficient funds, but the goods or services were already delivered to the customer.

Endpoint

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

Required Fields for Processing a Merchant-Initiated Resubmitted Transaction

Use these required fields to process a merchant-initiated resubmitted transaction.

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.phoneNumber

orderInformation.billTo.postalCode

paymentInformation.card.expirationMonth

paymentInformation.card.expirationYear

paymentInformation.card.number

processingInformation.authorizationOptions.initiator.merchantInitiatedTransaction.previousTransactionId	**For Discover and American Express cards, use the transaction ID from the original transaction. For Visa, use the last successful transaction ID.**

processingInformation. authorizationOptions. initiator. merchantInitiatedTransaction. reason	**Set the value to 1.**
processingInformation. authorizationOptions. initiator. type	**Set the value to merchant.**

Card-Specific Required Field for Processing a Merchant-Initiated Transactions

Discover

The listed card requires an additional field:

processingInformation. authorizationOptions. initiator. merchantInitiatedTransaction. originalAuthorizedAmount	**Provide the original transaction amount.**

REST Example: Processing a Merchant-Initiated Resubmitted Transaction

Endpoint:

- Production: POST https://api.cybersource.com/pts/v2/payments
- Test: POST https://apitest.cybersource.com/pts/v2/payments

Request

```
{
  "processingInformation": {
    "authorizationOptions": {
      "initiator": {
     "type": "merchant",
       "merchantInitiatedTransaction": {
        "originalAuthorizedAmount": "100",  // Discover Only
        "previousTransactionId": "123456789619999",
        "reason": "1"
       }
      }
    }
  },
  "orderInformation": {
 "billTo" : {
   "country" : "US",
   "lastName" : "Kim",
   "address1" : "201 S. Division St.",
   "postalCode" : "48104-2201",
   "locality" : "Ann Arbor",
   "administrativeArea" : "MI",
   "firstName" : "Kyong-Jin",
       "phoneNumber": "5554327113",
   "email" : "test@cybs.com"
```

```
    },
      "amountDetails": {
        "totalAmount": "100.00",
        "currency": "USD"
      }
    },
    "paymentInformation": {
      "card": {
        "expirationYear": "2031",
        "number": "4111xxxxxxxxxxx",
        "expirationMonth": "12"
      }
    }
}
```

Response to a Successful Request

```
{
    "_links": {
      "authReversal": {
        "method": "POST",
        "href": "/pts/v2/payments/6534232293716260503006/reversals"
      },
      "self": {
        "method": "GET",
        "href": "/pts/v2/payments/653423229371626050300 6"
      },
      "capture": {
        "method": "POST",
        "href": "/pts/v2/payments/6534232293716260503006/captures"
      }
    },
    "clientReferenceInformation": {
      "code": "1653423229353"
    },
    "id": "6534232293716260503006",
    "orderInformation": {
      "amountDetails": {
        "authorizedAmount": "100.00",
        "currency": "USD"
      }
    },
    "paymentAccountInformation": {
      "card": {
        "type": "004"
      }
    },
    "paymentInformation": {
      "tokenizedCard": {
        "type": "004"
      },
      "card": {
        "type": "004"
      }
    },
    "pointOfSaleInformation": {
```

```
      "terminalId": "111111"
    },
    "processorInformation": {
      "approvalCode": "888888",
      "networkTransactionId": "123456789619999",
      "transactionId": "123456789619999",
      "responseCode": "100",
      "avs": {
        "code": "X",
        "codeRaw": "I1"
      }
    },
    "reconciliationId": "64365912G3K7HFDJ",
    "status": "AUTHORIZED",
    "submitTimeUtc": "2022-05-24T20:13:49Z"
}
```


Merchant-Initiated Resubmission Transaction with TMS

A resubmission transaction is used when a merchant resubmits an authorization to recover an outstanding debt from the customer. A common scenario is when a card was initially declined due to insufficient funds, but the goods or services were already delivered to the customer.

This section describes how to process a merchant-initiated resubmission transaction using these TMS token types:

Customer	Customer tokens store one or more customer payment instrument tokens and shipping address tokens. Including a customer token eliminates the need to include billing information, card information, and the previous transaction's ID. `"paymentInformation": { "customer": { "id": "07C9CA98022DA498E063A2598D0AA400" } }` For more information about this TMS token type, see *Customer Tokens* in the Token Management Service Developer Guide.
Payment Instrument	Payment instrument tokens store an instrument identifier token, card information, and billing information.

	Payment instruments are not linked to a customer token. Including a payment instrument eliminates the need to include billing information, card information, and the previous transaction's ID. `"paymentInformation": { "paymentInstrument": { "id": "07CA24EF20F9E2C9E063A2598D0A8565" } }` For more information about this TMS token type, see *Payment Instrument Token* in the Token Management Service Developer Guide.
Instrument Identifier	Instrument identifier tokens store only a PAN. Including an instrument identifier eliminates the need to include a PAN and the previous transaction's ID. `"paymentInformation": { "instrumentIdentifier": { "id": "7010000000016241111" } }` For more information about this TMS token type, see *Instrument Identifier Token* in the Token Management Service Developer Guide.

Endpoint

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

Required Fields for MIT Resubmission Transaction with TMS

Include these Required Fields

orderInformation.amountDetails.currency

orderInformation.amountDetails.totalAmount

paymentInformation.[tokentype].id	Where **[tokentype]** is the TMS token type you are using: • **customer**

	• **instrumentIdentifier** • **paymentInstrument**
processingInformation. authorizationOptions. initiator. merchantInitiatedTransaction. reason	Set the value to 1.

Instrument Identifier Required Fields

If you are using the **paymentInformation.instrumentIdentifier.id** token, include these required fields in addition to the required fields listed above.

orderInformation.billTo.address1

orderInformation.billTo.administrativeArea

orderInformation.billTo.country

orderInformation.billTo.email

orderInformation.billTo.firstName

orderInformation.billTo.lastName

orderInformation.billTo.locality

orderInformation.billTo.phoneNumber

orderInformation.billTo.postalCode

paymentInformation.card.expirationMonth

paymentInformation.card.expirationYear

Card-Specific Fields

Include these fields when processing an authorization with these card types. The listed card type requires an additional field.

Diners Club	**processorInformation.cardReferenceData:** **Required only for token transactions. Set this field to the processorInformation.cardReferenceData field that was in the response message when you obtained the customer's credentials.** **issuerInformation.transactionInformation:** **Required only for token transactions. Set this field to the processorInformation.transactionID field that was in the response message when you obtained the customer's credentials.**

Discover	**processingInformation.authorizationOptions.initiator.merchantInitiatedTransaction.originalAuthorizedAmou** **Set to the original transaction amount.** **processorInformation.cardReferenceData** **Required only for token transactions. Set this field to the processorInformation.cardReferenceData field that was in the response message when you obtained the customer's credentials.** **issuerInformation.transactionInformation** **Required only for token transactions. Set this field to the processorInformation.transactionID field that was in the response message when you obtained the customer's credentials.**

Example: MIT Resubmission Transaction with a TMSInstrument Identifier

Endpoint:

- Production: POST https://api.cybersource.com/pts/v2/payments
- Test: POST https://apitest.cybersource.com/pts/v2/payments

Request

```
{
 "processingInformation": {
  "authorizationOptions": {
   "initiator": {
    "merchantInitiatedTransaction": {
     "reason": "1"
    }
   }
  }
 },
 "paymentInformation": {
  "card": {
   "expirationMonth": "12",
   "expirationYear": "2031"
  },
  "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"
    }
   }
}
```

Response to a Successful Request

```
{
 "_links": {
   "authReversal": {
    "method": "POST",
    "href": "/pts/v2/payments/6976922830456934003954/reversals"
   },
   "self": {
    "method": "GET",
    "href": "/pts/v2/payments/697692283045693400 3954"
   },
   "capture": {
    "method": "POST",
    "href": "/pts/v2/payments/6976922830456934003954/captures"
   }
 },
 "clientReferenceInformation": {
   "code": "1697692283160"
 },
 "id": "6976922830456934003954",
 "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"
  },
  "processingInformation": {
   "paymentSolution": "015"
  },
  "processorInformation": {
   "paymentAccountReferenceNumber": "V001001302229816966750 4231315",
   "approvalCode": "888888",
   "networkTransactionId": "123456789619999",
   "transactionId": "123456789619999",
   "responseCode": "100",
   "avs": {
    "code": "X",
    "codeRaw": "I1"
   }
  },
  "reconciliationId": "62700184NNMR6XFK",
  "status": "AUTHORIZED",
  "submitTimeUtc": "2023-10-19T05:11:23Z"
}
```


Example: MIT Resubmission Transaction with a TMSPayment Instrument

Endpoint:

- Production: POST https://api.cybersource.com/pts/v2/payments
- Test: POST https://apitest.cybersource.com/pts/v2/payments

Request


```
{
  "processingInformation": {
   "authorizationOptions": {
    "initiator": {
     "merchantInitiatedTransaction": {
      "reason": "1"
     }
    }
   }
  },
  "paymentInformation": {
   "paymentInstrument": {
    "id": "080AE120369A7947E063A2598D0A718F"
   }
  },
  "orderInformation": {
   "amountDetails": {
    "totalAmount": "102.21",
    "currency": "USD"
   }
```

```
  }
 }
```

Response to a Successful Request

```
{
  "_links": {
    "authReversal": {
      "method": "POST",
      "href": "/pts/v2/payments/6976917718796256603955/reversals"
    },
    "self": {
      "method": "GET",
      "href": "/pts/v2/payments/697691771879625603955"
    },
    "capture": {
      "method": "POST",
      "href": "/pts/v2/payments/6976917718796256603955/captures"
    }
  },
  "clientReferenceInformation": {
    "code": "1697691771976"
  },
  "id": "6976917718796256603955",
  "orderInformation": {
    "amountDetails": {
      "authorizedAmount": "102.21",
      "currency": "USD"
    }
  },
  "paymentAccountInformation": {
    "card": {
      "type": "001"
    }
  },
  "paymentInformation": {
    "tokenizedCard": {
      "type": "001"
    },
    "instrumentIdentifier": {
      "id": "7010000000016241111",
      "state": "ACTIVE"
    },
    "paymentInstrument": {
      "id": "080AE120369A7947E063A2598D0A718F"
    },
    "card": {
      "type": "001"
    }
  },
  "pointOfSaleInformation": {
    "terminalId": "111111"
  },
  "processingInformation": {
    "paymentSolution": "015"
  },
```

```
  "processorInformation": {
   "paymentAccountReferenceNumber": "V0010013022298169667504231315",
   "approvalCode": "888888",
   "networkTransactionId": "123456789619999",
   "transactionId": "123456789619999",
   "responseCode": "100",
   "avs": {
    "code": "X",
    "codeRaw": "I1"
   }
  },
  "reconciliationId": "62700629BNN13VGW",
  "status": "AUTHORIZED",
  "submitTimeUtc": "2023-10-19T05:02:52Z"
}
```


Example: MIT Reauthorization Transaction with a TMS Customer

Endpoint:

- Production: POST https://api.cybersource.com/pts/v2/payments
- Test: POST https://apitest.cybersource.com/pts/v2/payments

Request

```
{
 "processingInformation": {
  "authorizationOptions": {
   "initiator": {
    "merchantInitiatedTransaction": {
     "reason": "1"
    }
   }
  }
 },
 "paymentInformation": {
  "customer": {
   "id": "080AC9AB60C92AA2E063A2598D0A0C74"
  }
 },
 "orderInformation": {
  "amountDetails": {
   "totalAmount": "102.21",
   "currency": "USD"
  }
 }
}
```

Response to a Successful Request


```
{
 "_links": {
  "authReversal": {
   "method": "POST",
   "href": "/pts/v2/payments/6976916433716228003955/reversals"
```

```
    },
    "self": {
      "method": "GET",
      "href": "/pts/v2/payments/6976916433716228003955"
    },
    "capture": {
      "method": "POST",
      "href": "/pts/v2/payments/6976916433716228003955/captures"
    }
  },
  "clientReferenceInformation": {
    "code": "1697691643458"
  },
  "id": "6976916433716228003955",
  "orderInformation": {
    "amountDetails": {
      "authorizedAmount": "102.21",
      "currency": "USD"
    }
  },
  "paymentAccountInformation": {
    "card": {
      "type": "001"
    }
  },
  "paymentInformation": {
    "tokenizedCard": {
      "type": "001"
    },
    "instrumentIdentifier": {
      "id": "7010000000016241111",
      "state": "ACTIVE"
    },
    "paymentInstrument": {
      "id": "080AE6DB37B09557E063A2598D0AA4C9"
    },
    "card": {
      "type": "001"
    },
    "customer": {
      "id": "080AC9AB60C92AA2E063A2598D0A0C74"
    }
  },
  "pointOfSaleInformation": {
    "terminalId": "111111"
  },
  "processingInformation": {
    "paymentSolution": "015"
  },
  "processorInformation": {
    "paymentAccountReferenceNumber": "V0010013022298169667504231315",
    "approvalCode": "888888",
    "networkTransactionId": "123456789619999",
    "transactionId": "123456789619999",
    "responseCode": "100",
    "avs": {
```

```
    "code": "X",
    "codeRaw": "I1"
   }
  },
  "reconciliationId": "62700435FNN143RY",
  "status": "AUTHORIZED",
  "submitTimeUtc": "2023-10-19T05:00:43Z"
}
```


Installment Payments

An installment payment is a single purchase of goods or services billed to a customer in multiple transactions over a period of time agreed to by you and the customer. The agreement enables you to charge a specific amount at specified intervals.

Installments Service for Installment Payments

Important

Do not use this document if you are using the Installments service. When using the Installments service, Cybersource saves and stores payment credentials for installment transactions, ensuring compliance with COF best practices.

Customer-Initiated Installment Payments with PAN

An installment payment is a single purchase of goods or services billed to a customer in multiple transactions over a period of time agreed to by you and the customer. The agreement enables you to charge a specific amount at specified intervals.

Important

Do not use this document if you are using the Installments service. When using the Installments service, Cybersource saves and stores payment credentials for installment transactions, ensuring compliance with COF best practices.

Prerequisites

The first transaction in an installment payment is a customer-initiated transaction (CIT). Before you can perform a subsequent merchant-initiated transaction (MIT), you must store the customer's credentials for later use. Before you can store the user's credentials, you must get the customer's consent to store their private information. This process is also known as establishing a relationship with the customer.

Endpoint

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

Successful Response

Store the **processorInformation.networkTransactionId** field value from the successful response message. You must include the network transaction ID in subsequent MIT authorization requests to associate the CIT to the MIT.

Required Fields for Initial Customer-Initiated Installment Payments with a PAN

Include these required fields to authorize an initial customer-initiated installment payment using a PAN.

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.phoneNumber

orderInformation.billTo.postalCode

paymentInformation.card.expirationMonth

paymentInformation.card.expirationYear

paymentInformation.card.number

processingInformation. authorizationOptions. initiator. credentialStoredOnFile	**Set the value to `true`.**
processingInformation. authorizationOptions. initiator. type	**Set the value to `customer`.**
processingInformation. commerceIndicator	**Set the value to `internet` or a payer authentication value.**

Card-Specific Fields for Authorizing Initial Installment Payments

Use this required field if you are authorizing an initial installment payment using the card type referenced below.

Mastercard	**processingInformation. authorizationOptions. initiator. merchantInitiatedTransaction. reason** **Set the value to `9`.**

REST Example: Authorizing Initial Customer-Initiated Installment Payments with a PAN

Endpoint:

- Production: POST https://api.cybersource.com/pts/v2/payments
- Test: POST https://apitest.cybersource.com/pts/v2/payments

Request

```
{
  "processingInformation": {
    "commerceIndicator": "internet",
    "authorizationOptions": {
      "initiator": {
        "type": "customer",
        "credentialStoredOnFile": "true",
        "merchantInitiatedTransaction": {
          "reason": "9"   //Mastercard only
        }
      }
    }
  },
  "orderInformation": {
    "billTo": {
      "firstName": "John",
      "lastName": "Doe",
      "address1": "201 S. Division St.",
      "postalCode": "48104-2201",
      "locality": "Ann Arbor",
      "administrativeArea": "MI",
      "country": "US",
      "phoneNumber": "5554327113",
      "email": "test@cybs.com"
    },
    "amountDetails": {
      "totalAmount": "100.00",
      "currency": "USD"
    }
  },
  "paymentInformation": {
    "card": {
      "expirationYear": "2031",
      "number": "4111xxxxxxxxxxx",
      "expirationMonth": "12"
    }
  }
}
```

Response to a Successful Request

```
{
  "_links": {
    "authReversal": {
      "method": "POST",
      "href": "/pts/v2/payments/6528187198946076303004/reversals"
```

```
      },
      "self": {
        "method": "GET",
        "href": "/pts/v2/payments/652818719894607630304"
      },
      "capture": {
        "method": "POST",
        "href": "/pts/v2/payments/652818719894607630304/captures"
      }
    },
    "clientReferenceInformation": {
      "code": "1652818719876"
    },
    "id": "652818719894607630304",
    "orderInformation": {
      "amountDetails": {
        "authorizedAmount": "100.00",
        "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": "63165088Z3AHV91G",
    "status": "AUTHORIZED",
    "submitTimeUtc": "2022-05-17T20:18:40Z"
}
```

Customer-Initiated Installment Payment with TMS

An installment payment is a single purchase of goods or services billed to a customer in multiple transactions over a period of time agreed to by you and the customer. The agreement enables you to charge a specific amount at specified intervals.

> **Important**
>
> Do not use this document if you are using the Installments service. When using the Installments service, Cybersource saves and stores payment credentials for installment transactions, ensuring compliance with COF best practices.

Prerequisites

The first transaction in an installment payment is a customer-initiated transaction (CIT). Before you can perform a subsequent merchant-initiated transaction (MIT), you must store the customer's credentials for later use. Before you can store the user's credentials, you must get the customer's consent to store their private information. This process is also known as establishing a relationship with the customer.

Creating a TMS Token

When sending the initial CIT, you can create a TMS token to store the customer's credentials for the subsequent MITs. To create a TMS token, include the **processingInformation.actionTokenTypes** field in the authorization request. Set the field to one of these values based on the TMS token type you want to create:

Customer	**Customer tokens store one or more customer payment instrument tokens and shipping address tokens.** **Including a customer token in subsequent MITs eliminates the need to include billing information, card information, and the previous transaction's ID.** `"processingInformation": { "actionTokenTypes": [ "customer" ]` **For more information about this TMS token type, see *Customer Tokens* in the Token Management Service Developer Guide.**
Payment Instrument	**Payment instrument tokens store an instrument identifier token, card information, and billing information. Payment instruments are not linked to a customer token. Including a payment instrument in subsequent MITs eliminates**

	the need to include billing information, card information, and the previous transaction's ID. `"processingInformation": { "actionTokenTypes": [ "paymentInstrument" ]` **For more information about this TMS token type, see *Payment Instrument Token* in the Token Management Service Developer Guide.**
Instrument Identifier	**Instrument identifier tokens store a PAN. Including an instrument identifier in subsequent MITs eliminates the need to include a PAN and the previous transaction's ID.** `"processingInformation": { "actionTokenTypes": [ "instrumentIdentifier" ]` **For more information about this TMS token type, see *Instrument Identifier Token* in the Token Management Service Developer Guide.**
Instrument Identifier, Payment Instrument, and Customer Identifier	**You can also create multiple TMS token types in the same authorization. This example includes an instrument identifier, a payment instrument, and a customer token in the same authorization:** `"processingInformation": { "actionTokenTypes": [ "instrumentIdentifier", "paymentInstrument", "customer" ]`

Endpoint

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

Required Fields for CIT Installment Payments with TMS

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.phoneNumber

orderInformation.billTo.postalCode

paymentInformation.card.expirationMonth

paymentInformation.card.expirationYear

paymentInformation.card.number

processingInformation.actionList	**Set the value to** `TOKEN_CREATE`**.**
processingInformation.actionTokenTypes	**Set to one or more of these values:** • `customer` • `instrumentIdentifier` • `paymnentInstrument`
processingInformation.commerceIndicator	**Set the value to** `internet`**.**

Card-Specific Fields for Authorizing Initial Installment Payments

Use this required field if you are authorizing an initial installment payment using the card type referenced below.

Mastercard	**processingInformation. authorizationOptions. initiator. merchantInitiatedTransaction. reason** **Set the value to** `9`**.**

Related information
- API field reference guide for the REST API

REST Example: CIT Installment Payment with TMS

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

Request


```
{
 "processingInformation": {
  "actionList": [
   "TOKEN_CREATE"
  ],
  "actionTokenTypes": [
   "instrumentIdentifier"
  ],
  "commerceIndicator": "internet"
 },
 "paymentInformation": {
  "card": {
   "number": "411111111111XXXX",
   "expirationMonth": "12",
   "expirationYear": "2031"
  }
 },
 "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"
  }
 }
}
```


Response to a Successful Request


```
{
 "_links": {
  "authReversal": {
   "method": "POST",
   "href": "/pts/v2/payments/6972267090226779103955/reversals"
  },
  "self": {
   "method": "GET",
   "href": "/pts/v2/payments/6972267090226779103955"
  },
  "capture": {
   "method": "POST",
   "href": "/pts/v2/payments/6972267090226779103955/captures"
  }
 },
 "clientReferenceInformation": {
```

```
    "code": "TC50171_3"
   },
   "id": "6972267090226779103955",
   "orderInformation": {
    "amountDetails": {
     "authorizedAmount": "102.21",
     "currency": "USD"
    }
   },
   "paymentAccountInformation": {
    "card": {
     "type": "001"
    }
   },
   "paymentInformation": {
    "tokenizedCard": {
     "type": "001"
    },
    "card": {
     "type": "001"
    }
   },
   "pointOfSaleInformation": {
    "terminalId": "111111"
   },
   "processorInformation": {
    "paymentAccountReferenceNumber": "V0010013022298169667504231315",
    "approvalCode": "888888",
    "networkTransactionId": "123456789619999",
    "transactionId": "123456789619999",
    "responseCode": "100",
    "avs": {
     "code": "X",
     "codeRaw": "I1"
    }
   },
   "reconciliationId": "62506622XNMR6Q1Y",
   "status": "AUTHORIZED",
   "submitTimeUtc": "2023-10-13T19:51:49Z",
   "tokenInformation": {
    "instrumentidentifierNew": false,
    "instrumentIdentifier": {
     "state": "ACTIVE",
     "id": "7010000000016241111"
    }
   }
}
```


Merchant-Initiated Installment Payments with PAN

After the initial CIT installment payment, subsequent installment payments are merchant-initiated transactions (MITs).

Endpoint

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

Required Fields for Authorizing Merchant-Initiated Subsequent Installment Payments

Use these required fields to authorize merchant-initiated subsequent installment payments.

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.phoneNumber	
orderInformation.billTo.postalCode	
paymentInformation.card.expirationMonth	
paymentInformation.card.expirationYear	
paymentInformation.card.number	
processingInformation.authorizationOptions.initiator. merchantInitiatedTransaction. previousTransactionID	**For Discover and American Express cards, use the transaction ID from the original transaction. For Visa, use the last successful transaction ID.**
processingInformation. authorizationOptions. initiator. storedCredentialUsed	**Set the value to `true`.**
processingInformation. authorizationOptions. initiator. type	**Set the value to `merchant`.**
processingInformation. commerceIndicator	**Set the value to `install`.**
processingInformation. authorizationOptions. initiator. merchantInitiatedTransaction. reason	**Set the value to `9`.**  **Important**

	Only required for Mastercard transactions.

Card-Specific Required Field for Retrieving Customer Credentials During a CIT

Some card companies require additional information when making authorizations with stored credentials.

Discover

Discover requires the authorization amount from the original transaction when sending a request:

processingInformation.authorizationOptions.initiator.merchantInitiatedTransaction.originalAuthorizedAmount

India-Specific Required Fields for Installment Payments

This section shows the required fields for Diners Club, Mastercard, and Visa in India.

Diners Club and Mastercard

Use these fields for authorizing an MIT installment payment when processing payments through Visa Platform Connect.

installmentInformation.amount	
installmentInformation.frequency	**Required only for the first MIT installment payment.**
installmentInformation.identifier	
installmentInformation.paymentType	
installmentInformation.sequence	
installmentInformation.validIndicator	

Visa

Use this field for authorizing a MIT installment payment when processing payments through Visa Platform Connect.

installmentInformation.identifier

REST Example: Authorizing Merchant-Initiated Subsequent Installment Payments

Endpoint:

- Production: POST https://api.cybersource.com/pts/v2/payments
- Test: POST https://apitest.cybersource.com/pts/v2/payments

Request

```
{
  "processingInformation": {
    "commerceIndicator": "install",
    "authorizationOptions": {
      "initiator": {
        "storedCredentialUsed": "true",
        "type": "merchant",
        "merchantInitiatedTransaction": {
          "reason": "9",
          "previousTransactionId": "123456789619999",
          "originalAuthorizedAmount": "100"    //Discover Only
        }
      }
    }
  },
  "orderInformation": {
    "billTo": {
      "firstName": "John",
      "lastName": "Doe",
      "address1": "201 S. Division St.",
      "postalCode": "48104-2201",
      "locality": "Ann Arbor",
      "administrativeArea": "MI",
      "country": "US",
      "phoneNumber": "5554327113",
      "email": "test@cybs.com"
    },
    "amountDetails": {
      "totalAmount": "100.00",
      "currency": "USD"
    }
  },
  "paymentInformation": {
    "card": {
      "expirationYear": "2031",
      "number": "4111xxxxxxxxxxx",
      "expirationMonth": "12"
    }
  }
}
```

Response to a Successful Request

```
{
  "_links": {
    "authReversal": {
      "method": "POST",
      "href": "/pts/v2/payments/6530824710046809304002/reversals"
    },
    "self": {
      "method": "GET",
      "href": "/pts/v2/payments/653082471004680930400 2"
    },
    "capture": {
```

```
        "method": "POST",
        "href": "/pts/v2/payments/6530824710046809304002/captures"
      }
    },
    "clientReferenceInformation": {
      "code": "1653082470983"
    },
    "id": "6530824710046809304002",
    "orderInformation": {
      "amountDetails": {
        "authorizedAmount": "100.00",
        "currency": "USD"
      }
    },
    "paymentAccountInformation": {
      "card": {
        "type": "002"
      }
    },
    "paymentInformation": {
      "tokenizedCard": {
        "type": "002"
      },
      "card": {
        "type": "002"
      }
    },
    "pointOfSaleInformation": {
      "terminalId": "111111"
    },
    "processorInformation": {
      "approvalCode": "888888",
      "authIndicator": "1",
      "networkTransactionId": "123456789619999",
      "transactionId": "123456789619999",
      "responseCode": "100",
      "avs": {
        "code": "X",
        "codeRaw": "I1"
      }
    },
    "reconciliationId": "79710341A39WTT5W",
    "status": "AUTHORIZED",
    "submitTimeUtc": "2022-05-20T21:34:31Z"
}
```


Merchant-Initiated Installment Payment with TMS

This section describes how to process a merchant-initiated installment payment using these TMS token types:

Customer	Customer tokens store one or more customer payment instrument tokens and shipping address tokens.

	Including a customer token eliminates the need to include billing information, card information, and the previous transaction's ID.
	
	`"paymentInformation": {`
	`"customer": {`
	`"id": "07C9CA98022DA498E063A2598D0AA400"`
	`}`
	`}`
	
	For more information about this TMS token type, see *Customer Tokens* in the Token Management Service Developer Guide.

Payment Instrument	Payment instrument tokens store an instrument identifier token, card information, and billing information. Payment instruments are not linked to a customer token.
	
	Including a payment instrument eliminates the need to include billing information, card information, and the previous transaction's ID.
	
	`"paymentInformation": {`
	`"paymentInstrument": {`
	`"id": "07CA24EF20F9E2C9E063A2598D0A8565"`
	`}`
	`}`
	
	For more information about this TMS token type, see *Payment Instrument Token* in the Token Management Service Developer Guide.

Instrument Identifier	Instrument identifier tokens store only a PAN. Including an instrument identifier eliminates the need to include a PAN and the previous transaction's ID.
	
	`"paymentInformation": {`
	`"instrumentIdentifier": {`
	`"id": "7010000000016241111"`
	`}`
	`}`
	
	For more information about this TMS token type, see *Instrument Identifier Token* in the Token Management Service Developer Guide.

Endpoint

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

Required Fields for MIT Installment Payments with TMS

Include these Required Fields

orderInformation.amountDetails.currency

orderInformation.amountDetails.totalAmount

paymentInformation.[tokentype].id	Where **[tokentype]** is the TMS token type you are using: • **customer** • **instrumentIdentifier** • **paymentInstrument**
processingInformation.commerceIndicator	Set the value to `install`.

Instrument Identifier Required Fields

If you are using the **paymentInformation.instrumentIdentifier.id** token, include these required fields in addition to the required fields listed above.

orderInformation.billTo.address1

orderInformation.billTo.administrativeArea

orderInformation.billTo.country

orderInformation.billTo.email

orderInformation.billTo.firstName

orderInformation.billTo.lastName

orderInformation.billTo.locality

orderInformation.billTo.phoneNumber

orderInformation.billTo.postalCode

paymentInformation.card.expirationMonth

paymentInformation.card.expirationYear

Card-Specific Required Field for Retrieving Customer Credentials During a CIT

Some card companies require additional information when making authorizations with stored credentials.

Discover

Discover requires the authorization amount from the original transaction when sending a request:

processingInformation.authorizationOptions.initiator.merchantInitiatedTransaction.originalAuthorizedAmount

India-Specific Required Fields for Installment Payments

This section shows the required fields for Diners Club, Mastercard, and Visa in India.

Diners Club and Mastercard

Use these fields for authorizing an MIT installment payment when processing payments through Visa Platform Connect.

installmentInformation.amount	
installmentInformation.frequency	**Required only for the first MIT installment payment.**
installmentInformation.identifier	
installmentInformation.paymentType	
installmentInformation.sequence	
installmentInformation.validIndicator	

Visa

Use this field for authorizing a MIT installment payment when processing payments through Visa Platform Connect.

installmentInformation.identifier

Example: MIT with TMS Instrument Identifier Token

Request

Endpoint: POST https://api.cybersource.com/pts/v2/payments

```
{
 "processingInformation": {
  "commerceIndicator": "install"
 },
 "paymentInformation": {
  "card": {
   "expirationMonth": "12",
   "expirationYear": "2031"
  },
  "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"
    }
  }
}
```

Response to a Successful Request

```
{
    "_links": {
        "authReversal": {
            "method": "POST",
            "href": "/pts/v2/payments/6530824710046809304002/reversals"
        },
        "self": {
            "method": "GET",
            "href": "/pts/v2/payments/653082471004680930402"
        },
        "capture": {
            "method": "POST",
            "href": "/pts/v2/payments/6530824710046809304002/captures"
        }
    },
    "clientReferenceInformation": {
        "code": "1653082470983"
    },
    "id": "6530824710046809304002",
    "orderInformation": {
        "amountDetails": {
            "authorizedAmount": "100.00",
            "currency": "USD"
        }
    },
    "paymentAccountInformation": {
        "card": {
            "type": "002"
        }
    },
    "paymentInformation": {
        "tokenizedCard": {
            "type": "002"
        },
        "card": {
```

```
        "type": "002"
      }
    },
    "pointOfSaleInformation": {
      "terminalId": "111111"
    },
    "processorInformation": {
      "approvalCode": "888888",
      "authIndicator": "1",
      "networkTransactionId": "123456789619999",
      "transactionId": "123456789619999",
      "responseCode": "100",
      "avs": {
        "code": "X",
        "codeRaw": "I1"
      }
    },
    "reconciliationId": "79710341A39WTT5W",
    "status": "AUTHORIZED",
    "submitTimeUtc": "2022-05-20T21:34:31Z"
}
```


Recurring Payments

A recurring payment is a credentials-on-file (COF) transaction in a series of payments that you bill to a customer for a fixed amount at regular intervals that do not exceed one year between transactions. The series of recurring payments is the result of an agreement between you and the customer for the purchase of goods or services that are provided at regular intervals.
Mastercard uses standing order and subscription payments instead of recurring payments. See *Mastercard Standing Order Payments* on page 386 and *Mastercard Subscription Payments* on page 393.

Recurring Billing Service for Recurring Payments

Important

Do not use this document for the Recurring Billing service.
Use the *Recurring Billing Developer Guide*. When you use the Recurring Billing service, Cybersource saves and stores payment credentials for recurring transactions, ensuring compliance with COF best practices.

Customer-Initiated Recurring Payment with PAN

A recurring payment is a credentials-on-file (COF) transaction in a series of payments that you bill to a customer at a fixed amount, at regular intervals that do not exceed one year between transactions. The series of recurring payments is the result of an agreement

between you and the customer for the purchase of goods or services that are provided at regular intervals.
Mastercard uses standing order and subscription payments instead of recurring payments. See *Mastercard Standing Order Payments* on page 386 and *Mastercard Subscription Payments* on page 393.

Prerequisites

The first transaction in a recurring payment is a customer-initiated transaction (CIT). Before you can perform a subsequent merchant-initiated transaction (MIT), you must store the customer's credentials for later use. Before you can store the customer's credentials, you must get their consent to store their private information. This is also known as establishing a relationship with the customer.

Recurring Billing Service for Recurring Payments

Important

Do not use this document for the Recurring Billing service.
Use the *Recurring Billing Developer Guide*. When you use the Recurring Billing service, Cybersource saves and stores payment credentials for recurring transactions, ensuring compliance with COF best practices.

Endpoint

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

Successful Response

Store the **processorInformation.networkTransactionId** field value from the successful response message. You must include the network transaction ID in subsequent MIT authorization requests to associate the CIT to the MIT.

Required Fields for Authorizing a Customer-Initiated Recurring Payment with PAN Using REST API

Use these required fields to request an initial customer-initiated recurring payment.

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	
processingInformation. authorizationOptions. initiator. credentialStoredOnFile	**Set the value to `true`.**
processingInformation. authorizationOptions. initiator. type	**Set the value to `customer`.**
processingInformation. commerceIndicator	**Set the value to `internet`, a payer authentication value, or `MOTO`.**
processingInformation. recurringOptions. firstRecurringPayment	**Set the value to `true`.**

REST Example: Authorizing a Customer-Initiated Recurring Payment with a PAN

Use this REST API request for the initial customer-initiated recurring payment. In the response, find the transaction ID value in the **processorInformation.transactionId** field and save it for subsequent merchant-initiated recurring authorization requests. If you are using a token, the transaction ID is automatically stored.
Endpoint:

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

Request

```
{
  "processingInformation": {
    "commerceIndicator": "internet",
    "authorizationOptions": {
      "initiator": {
        "credentialStoredOnFile": "true",
        "type": "customer"
      }
    }
  },
  "orderInformation": {
    "billTo": {
      "firstName": "John",
      "lastName": "Doe",
      "address1": "201 S. Division St.",
      "postalCode": "48104-2201",
      "locality": "Ann Arbor",
```

```
        "administrativeArea": "MI",
        "country": "US",
        "phoneNumber": "5554327113",
        "email": "test@cybs.com"
      },
      "amountDetails": {
        "totalAmount": "100.00",
        "currency": "USD"
      }
    },
    "paymentInformation": {
      "card": {
        "expirationYear": "2031",
        "number": "4111xxxxxxxxxxx",
        "expirationMonth": "12"
      }
    }
}
```

Response to a Successful Request

```
{
    "_links": {
      "authReversal": {
        "method": "POST",
        "href": "/pts/v2/payments/6528187198946076303004/reversals"
      },
      "self": {
        "method": "GET",
        "href": "/pts/v2/payments/652818719894607630304"
      },
      "capture": {
        "method": "POST",
        "href": "/pts/v2/payments/6528187198946076303004/captures"
      }
    },
    "clientReferenceInformation": {
      "code": "1652818719876"
    },
    "id": "652818719894607630304",
    "orderInformation": {
      "amountDetails": {
        "authorizedAmount": "100.00",
        "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": "63165088Z3AHV91G",
    "status": "AUTHORIZED",
    "submitTimeUtc": "2022-05-17T20:18:40Z"
}
```


Customer-Initiated Recurring Payment with TMS

A recurring payment is a credentials-on-file (COF) transaction in a series of payments that you bill to a customer at a fixed amount, at regular intervals that do not exceed one year between transactions. The series of recurring payments is the result of an agreement between you and the customer for the purchase of goods or services that are provided at regular intervals.
Mastercard uses standing order and subscription payments instead of recurring payments. See *Mastercard Standing Order Payments* on page 386 and *Mastercard Subscription Payments* on page 393.

Prerequisites

The first transaction in a recurring payment is a customer-initiated transaction (CIT). Before you can perform a subsequent merchant-initiated transaction (MIT), you must store the customer's credentials for later use. Before you can store the customer's credentials, you must get their consent to store their private information. This is also known as establishing a relationship with the customer.

Recurring Billing Service for Recurring Payments

Important

Do not use this document for the Recurring Billing service.
Use the *Recurring Billing Developer Guide*. When you use the Recurring Billing service, Cybersource saves and stores payment credentials for recurring transactions, ensuring compliance with COF best practices.

Creating a TMS Token

When sending the initial CIT, you can create a TMS token to store the customer's credentials for the subsequent MITs. To create a TMS token, include the **processingInformation.actionTokenTypes** field in the authorization request. Set the field to one of these values based on the TMS token type you want to create:

Customer

Customer tokens store one or more customer payment instrument tokens and shipping address tokens.

Including a customer token in subsequent MITs eliminates the need to include billing information, card information, and the previous transaction's ID.

```
"processingInformation": {
  "actionTokenTypes": [
   "customer"
]
```

For more information about this TMS token type, see *Customer Tokens* in the Token Management Service Developer Guide.

Payment Instrument

Payment instrument tokens store an instrument identifier token, card information, and billing information. Payment instruments are not linked to a customer token. Including a payment instrument in subsequent MITs eliminates the need to include billing information, card information, and the previous transaction's ID.

```
"processingInformation": {
  "actionTokenTypes": [
   "paymentInstrument"
]
```

For more information about this TMS token type, see *Payment Instrument Token* in the Token Management Service Developer Guide.

Instrument Identifier

Instrument identifier tokens store a PAN. Including an instrument identifier in subsequent MITs eliminates the need to include a PAN and the previous transaction's ID.

```
"processingInformation": {
```

```
  "actionTokenTypes": [
   "instrumentIdentifier"
]
```

For more information about this TMS token type, see *Instrument Identifier Token* in the Token Management Service Developer Guide.

Instrument Identifier, Payment Instrument, and Customer Identifier

You can also create multiple TMS token types in the same authorization. This example includes an instrument identifier, a payment instrument, and a customer token in the same authorization:

```
"processingInformation": {
  "actionTokenTypes": [
     "instrumentIdentifier",
     "paymentInstrument",
     "customer"
]
```

Endpoint

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

Required Fields for Authorizing a Customer-Initiated Recurring Payment with TMS

Use these required fields to request a customer-initiated recurring payment with TMS.

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	
processingInformation.actionList	**Set the value to** `TOKEN_CREATE`.
processingInformation.actionTokenTypes	**Set to one or more of these values:** • `customer` • `instrumentIdentifier` • `paymentInstrument`
processingInformation.commerceIndicator	**Set the value to** `internet`.
processingInformation.recurringOptions.firstRecurringPayment	**Set the value to** `true`.

REST Example: Authorizing a Customer-Initiated Recurring Payment with TMS

Use this REST API request for the initial CIT recurring payment.
Endpoint:

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

Request

```
{
 "processingInformation": {
  "actionList": [
   "TOKEN_CREATE"
  ],
  "actionTokenTypes": [
   "customer"
  ],
  "commerceIndicator": "internet",
  "recurringOptions": {
   "firstRecurringPayment": true
  }
 },
 "paymentInformation": {
  "card": {
   "number": "4111111111111111",
   "expirationMonth": "12",
   "expirationYear": "2031"
  }
 },
 "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": ""
   }
  }
}
```

Response to a Successful Request

```
{
 "_links": {
  "authReversal": {
   "method": "POST",
   "href": "/pts/v2/payments/6976858134106105703954/reversals"
  },
  "self": {
   "method": "GET",
   "href": "/pts/v2/payments/6976858134106105703954"
  },
  "capture": {
   "method": "POST",
   "href": "/pts/v2/payments/6976858134106105703954/captures"
  }
 },
 "clientReferenceInformation": {
  "code": "1697685813462"
 },
 "id": "6976858134106105703954",
 "orderInformation": {
  "amountDetails": {
   "authorizedAmount": "102.21",
   "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": "62698397FNN143CC",
   "status": "AUTHORIZED",
   "submitTimeUtc": "2023-10-19T03:23:33Z",
   "tokenInformation": {
    "customer": {
     "id": "080A3A742BF87171E063A2598D0AEABE"
    }
   }
  }
```


Merchant-Initiated Recurring Payments with PAN

After the initial recurring payment (CIT), subsequent recurring payments are merchant-initiated transactions (MITs).

Endpoint

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

Required Fields for Authorizing a Merchant-Initiated Recurring Payment

Use these required fields to authorize subsequent recurring payments.

authorizationOptions. initiator. merchantInitiatedTransaction. agreementId	**Required for the first MIT recurring payment and subsequent MIT recurring payments if your business is located in Saudi Arabia.**
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.phoneNumber	
orderInformation.billTo.postalCode	
paymentInformation.card.expirationMonth	

paymentInformation.card.expirationYear	
paymentInformation. card. number	
processingInformation. authorizationOptions. initiator. merchantInitiatedTransaction. previousTransactionID	**For Discover and American Express cards, use the transaction ID from the original transaction. For Visa, use the last successful transaction ID.**
processingInformation. authorizationOptions. initiator. storedCredentialUsed	**Set the value to `true`.**
processingInformation. authorizationOptions. initiator. type	**Set the value to `merchant`.**
processingInformation. commerceIndicator	**Set the value to `recurring`.**

Card-Specific Required Fields for Authorizing Subsequent Recurring Payments

Some card companies require additional information when making authorizations with stored credentials.

Discover

Include the authorization amount from the original transaction in this field:

processingInformation.authorizationOptions.initiator.merchantInitiatedTransaction. originalAuthorizedAmount

Mastercard

Mastercard supports subscription and standing order payments instead of recurring payments. See *Mastercard Subscription Payments* on page 393 and *Mastercard Standing Order Payments* on page 386.

Country-Specific Required Fields for Authorizing Subsequent Recurring Payments

Include these country-specific required fields for a successful merchant-initiated authorization.

India

These fields are required only with Diners Club in India or with an India-issued card, and you are processing payments through Visa Platform Connect.

installmentInformation.amount

installmentInformation.frequency

installmentInformation.identifier

installmentInformation.paymentType

installmentInformation.sequence

installmentInformation.validationIndicator

Saudi Arabia

These fields are required only if your business is located in Saudi Arabia and you are processing payments through Visa Platform Connect.

authorizationOptions.initiator.merchantInitiatedTransaction.agreementId

recurringPaymentInformation.amountType

REST Example: Authorizing a Merchant-Initiated Recurring Payment

Endpoint:

- Production: POST https://api.cybersource.com/pts/v2/payments
- Test: POST https://apitest.cybersource.com/pts/v2/payments

Request

```
{
  "processingInformation": {
    "commerceIndicator": "recurring",
    "authorizationOptions": {
      "initiator": {
        "storedCredentialUsed": "true",
        "type": "merchant",
        "merchantInitiatedTransaction": {
          "reason": "7",
          "previousTransactionId": "123456789619999",
          "originalAuthorizedAmount": "100"    //Discover Only
        }
      }
    }
  },
  "orderInformation": {
    "billTo": {
      "firstName": "John",
      "lastName": "Doe",
      "address1": "201 S. Division St.",
      "postalCode": "48104-2201",
      "locality": "Ann Arbor",
      "administrativeArea": "MI",
      "country": "US",
      "phoneNumber": "5554327113",
      "email": "test@cybs.com"
    },
    "amountDetails": {
      "totalAmount": "100.00",
      "currency": "USD"
    }
  },
  "paymentInformation": {
    "card": {
      "expirationYear": "2031",
      "number": "4111xxxxxxxxxxx",
      "expirationMonth": "12"
    }
  }
```

```
}
```

Response to a Successful Request

```
{
  "_links": {
    "authReversal": {
      "method": "POST",
      "href": "/pts/v2/payments/6530824710046809304002/reversals"
    },
    "self": {
      "method": "GET",
      "href": "/pts/v2/payments/653082471004680930400 2"
    },
    "capture": {
      "method": "POST",
      "href": "/pts/v2/payments/6530824710046809304002/captures"
    }
  },
  "clientReferenceInformation": {
    "code": "1653082470983"
  },
  "id": "6530824710046809304002",
  "orderInformation": {
    "amountDetails": {
      "authorizedAmount": "100.00",
      "currency": "USD"
    }
  },
  "paymentAccountInformation": {
    "card": {
      "type": "002"
    }
  },
  "paymentInformation": {
    "tokenizedCard": {
      "type": "002"
    },
    "card": {
      "type": "002"
    }
  },
  "pointOfSaleInformation": {
    "terminalId": "111111"
  },
  "processorInformation": {
    "approvalCode": "888888",
    "authIndicator": "1",
    "networkTransactionId": "123456789619999",
    "transactionId": "123456789619999",
    "responseCode": "100",
    "avs": {
      "code": "X",
      "codeRaw": "I1"
    }
  },
```

```
  "reconciliationId": "79710341A39WTT5W",
  "status": "AUTHORIZED",
  "submitTimeUtc": "2022-05-20T21:34:31Z"
}
```

Merchant-Initiated Recurring Payments with TMS

After the customer-initiated recurring payment, you can send merchant-initiated recurring payments using one or more TMS token types:

Customer	Customer tokens store one or more customer payment instrument tokens and shipping address tokens. Including a customer token eliminates the need to include billing information, card information, and the previous transaction's ID. `"paymentInformation": { "customer": { "id": "07C9CA98022DA498E063A2598D0AA400" } }` For more information about this TMS token type, see *Customer Tokens* in the Token Management Service Developer Guide.
Payment Instrument	Payment instrument tokens store an instrument identifier token, card information, and billing information. Payment instruments are not linked to a customer token. Including a payment instrument eliminates the need to include billing information, card information, and the previous transaction's ID. `"paymentInformation": { "paymentInstrument": { "id": "07CA24EF20F9E2C9E063A2598D0A8565" } }` For more information about this TMS token type, see *Payment Instrument Token* in the Token Management Service Developer Guide.
Instrument Identifier	Instrument identifier tokens store only a PAN. Including an instrument identifier

eliminates the need to include a PAN and the previous transaction's ID.

```
"paymentInformation": {
 "instrumentIdentifier": {
  "id": "7010000000016241111"
 }
}
```

For more information about this TMS token type, see *Instrument Identifier Token* in the Token Management Service Developer Guide.

Endpoint

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

Required Fields for Authorizing a Merchant-Initiated Recurring Payments with TMS

Use these required fields to authorize subsequent recurring payments.

orderInformation.amountDetails.currency

orderInformation.amountDetails.totalAmount

paymentInformation.[tokentype].id

Where [tokentype] is the TMS token type you are using:

- **customer**
- **instrumentIdentifier**
- **paymentInstrument**

processingInformation.commerceIndicator

Set the value to `recurring`.

Instrument Identifier Required Fields

If you are using the **paymentInformation.instrumentIdentifier.id** token, include these required fields in addition to the required fields listed above.

orderInformation.billTo.address1

orderInformation.billTo.administrativeArea

orderInformation.billTo.country

orderInformation.billTo.email

orderInformation.billTo.firstName

orderInformation.billTo.lastName

orderInformation.billTo.locality

orderInformation.billTo.phoneNumber

orderInformation.billTo.postalCode

paymentInformation.card.expirationMonth

paymentInformation.card.expirationYear

Card-Specific Field

Some card companies require additional fields when making authorizations with stored credentials. Include this field if you are using these card types:

Discover	**processingInformation.authorizationOptions.initiator.merchantInitiatedTransaction.originalAuthorizedAmou**
Mastercard	**Mastercard supports subscription and standing order payments instead of recurring payments. See *Mastercard Subscription Payments* on page 393 and *Mastercard Standing Order Payments* on page 386.**

Country-Specific Field

Some countries require additional fields in order to process an authorization. Include this field if your business is located in this country:

Saudi Arabia	**authorizationOptions.initiator.merchantInitiatedTransa** **Required for the first MIT recurring payment and subsequent MIT recurring payments.**

REST Example: Authorizing a Merchant-Initiated Recurring Payment with a TMS Instrument Identifier

This example shows a successful merchant-initiated recurring payment using a TMS instrument identifier token.

Endpoint:

- Production: POST https://api.cybersource.com/pts/v2/payments
- Test: POST https://apitest.cybersource.com/pts/v2/payments

Request

```
{
  "processingInformation": {
    "commerceIndicator": "recurring"
  },
  "paymentInformation": {
    "card": {
      "expirationMonth": "12",
      "expirationYear": "2025"
```

```
    },
    "instrumentIdentifier": {
      "id": "4111xxxxxxxxxxx"
    }
  },
  "orderInformation": {
    "amountDetails": {
      "totalAmount": "102.21",
      "currency": "USD"
    },
    "billTo": {
      "firstName": "John",
      "lastName": "Smith",
      "address1": "1 Market St",
      "locality": "san francisco",
      "administrativeArea": "CA",
      "postalCode": "94105",
      "country": "US",
      "email": "test@cybs.com",
      "phoneNumber": "4158880000"
    }
  }
}
```

Response to a Successful Request

```
{
  "_links": {
    "authReversal": {
      "method": "POST",
      "href": "/pts/v2/payments/6530824710046809304002/reversals"
    },
    "self": {
      "method": "GET",
      "href": "/pts/v2/payments/653082471004680930402"
    },
    "capture": {
      "method": "POST",
      "href": "/pts/v2/payments/6530824710046809304002/captures"
    }
  },
  "clientReferenceInformation": {
    "code": "1653082470983"
  },
  "id": "6530824710046809304002",
  "orderInformation": {
    "amountDetails": {
      "authorizedAmount": "100.00",
      "currency": "USD"
    }
  },
  "paymentAccountInformation": {
    "card": {
      "type": "002"
    }
  },
```

```
    "paymentInformation": {
      "tokenizedCard": {
        "type": "002"
      },
      "card": {
        "type": "002"
      }
    },
    "pointOfSaleInformation": {
      "terminalId": "111111"
    },
    "processorInformation": {
      "approvalCode": "888888",
      "authIndicator": "1",
      "networkTransactionId": "123456789619999",
      "transactionId": "123456789619999",
      "responseCode": "100",
      "avs": {
        "code": "X",
        "codeRaw": "I1"
      }
    },
    "reconciliationId": "79710341A39WTT5W",
    "status": "AUTHORIZED",
    "submitTimeUtc": "2022-05-20T21:34:31Z"
}
```

REST Example: Authorizing a Merchant-Initiated Recurring Payment with TMS Payment Instrument

This example shows a successful merchant-initiated recurring payment using a TMS payment instrument token.

Endpoint:

- Production: POST https://api.cybersource.com/pts/v2/payments
- Test: POST https://apitest.cybersource.com/pts/v2/payments

Request

```
{
  "clientReferenceInformation": {
    "code": "TC50171_3"
  },
  "processingInformation": {
    "commerceIndicator": "recurring"
  },
  "paymentInformation": {
    "paymentInstrument": {
      "id": "07DB0915C20F2DDBE063A2598D0A6F26"
    }
  },
  "orderInformation": {
    "amountDetails": {
      "totalAmount": "102.21",
      "currency": "USD"
```

```
    }
  }
}
```

Response to a Successful Request

```
{
  "_links": {
    "authReversal": {
      "method": "POST",
      "href": "/pts/v2/payments/697483990810630410 3955/reversals"
    },
    "self": {
      "method": "GET",
      "href": "/pts/v2/payments/69748399081063041 03955"
    },
    "capture": {
      "method": "POST",
      "href": "/pts/v2/payments/697483990810630410 3955/captures"
    }
  },
  "clientReferenceInformation": {
    "code": "TC50171_3"
  },
  "id": "6974839908106304103955",
  "orderInformation": {
    "amountDetails": {
      "authorizedAmount": "102.21",
      "currency": "USD"
    }
  },
  "paymentAccountInformation": {
    "card": {
      "type": "001"
    }
  },
  "paymentInformation": {
    "tokenizedCard": {
      "type": "001"
    },
    "instrumentIdentifier": {
      "id": "7010000000016241111",
      "state": "ACTIVE"
    },
    "paymentInstrument": {
      "id": "07DB0915C20F2DDBE063A2598D0A6F26"
    },
    "card": {
      "type": "001"
    }
  },
  "pointOfSaleInformation": {
    "terminalId": "111111"
  },
  "processingInformation": {
    "paymentSolution": "015"
```

```
  },
  "processorInformation": {
    "paymentAccountReferenceNumber": "V0010013022298169667504231315",
    "approvalCode": "888888",
    "networkTransactionId": "123456789619999",
    "transactionId": "123456789619999",
    "responseCode": "100",
    "avs": {
      "code": "X",
      "codeRaw": "I1"
    }
  },
  "reconciliationId": "62599243NNMR6324",
  "status": "AUTHORIZED",
  "submitTimeUtc": "2023-10-16T19:19:51Z"
}
```


REST Example: Authorizing a Merchant-Initiated Recurring Payment with a TMS Customer Token

This example shows a successful merchant-initiated recurring payment using a TMS customer token.

Endpoint:

- Production: POST https://api.cybersource.com/pts/v2/payments
- Test: POST https://apitest.cybersource.com/pts/v2/payments

Request

```
{
  "clientReferenceInformation": {
    "code": "TC50171_3"
  },
  "processingInformation": {
    "commerceIndicator": "recurring"
  },
  "paymentInformation": {
    "customer": {
      "id": "07DB50E35AE11DA2E063A2598D0A9995"
    }
  },
  "orderInformation": {
    "amountDetails": {
      "totalAmount": "102.21",
      "currency": "USD"
    }
  }
}
```

Response to a Successful Request


```
{
  "_links": {
    "authReversal": {
      "method": "POST",
      "href": "/pts/v2/payments/6974846967476340503955/reversals"
```

```
    },
    "self": {
      "method": "GET",
      "href": "/pts/v2/payments/697484696747634050 3955"
    },
    "capture": {
      "method": "POST",
      "href": "/pts/v2/payments/697484696747634050 3955/captures"
    }
  },
  "clientReferenceInformation": {
    "code": "TC50171_3"
  },
  "id": "6974846967476340503955",
  "orderInformation": {
    "amountDetails": {
      "authorizedAmount": "102.21",
      "currency": "USD"
    }
  },
  "paymentAccountInformation": {
    "card": {
      "type": "001"
    }
  },
  "paymentInformation": {
    "tokenizedCard": {
      "type": "001"
    },
    "card": {
      "type": "001"
    }
  },
  "pointOfSaleInformation": {
    "terminalId": "111111"
  },
  "processingInformation": {
    "paymentSolution": "015"
  },
  "processorInformation": {
    "paymentAccountReferenceNumber": "V001001302229816966750 4231315",
    "approvalCode": "888888",
    "networkTransactionId": "123456789619999",
    "transactionId": "123456789619999",
    "responseCode": "100",
    "avs": {
      "code": "X",
      "codeRaw": "I1"
    }
  },
  "reconciliationId": "62599950BNN133LK",
  "status": "AUTHORIZED",
  "submitTimeUtc": "2023-10-16T19:31:36Z"
}
```

Mastercard Standing Order Payments

A standing order payment is a recurring COF transaction that is a variable amount at a regular interval, such as a utility bill, not to exceed one year between transactions. The series of recurring payments is the result of an agreement between you and the customer for the purchase of goods or services that are provided at regular intervals.

Mastercard Initial CIT Standing Order Payment

The first transaction in a standing order payment is a customer-initiated transaction (CIT). Before you can perform a subsequent merchant-initiated transaction (MIT), you must store the customer's credentials for later use. Before you can store the user's credentials, you must get the customer's consent to store their private information. This process is also known as establishing a relationship with the customer.

Endpoint

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

Successful Response

Store the **processorInformation.networkTransactionId** field value from the successful response message. You must include the network transaction ID in subsequent MIT authorization requests to associate the CIT to the MIT.

Required Fields for Authorizing Initial CIT Standing Order Payments

Use these required fields to authorize initial customer-initated standing order payments.

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.phoneNumber

orderInformation.billTo.postalCode

paymentInformation.card.expirationMonth

paymentInformation.card.expirationYear	
paymentInformation.card.number	
processingInformation. authorizationOptions. initiator. credentialStoredOnFile	Set the value to `true`.
processingInformation. authorizationOptions. initiator.type	Set the value to `customer`.
processingInformation. commerceIndicator	Set the value to `internet` or a payer authentication value.
processingInformation. authorizationOptions. initiator. merchantInitiatedTransaction. reason	Set the value to `8`.

REST Example: Authorizing Initial CIT Standing Order Payments

Endpoint:

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

Request

```
{
  "processingInformation": {
    "commerceIndicator": "internet",
    "authorizationOptions": {
      "initiator": {
        "credentialStoredOnFile": "true",
        "type": "customer",
        "merchantInitiatedTransaction": {
          "reason": "8"
        }
      }
    }
  },
  "orderInformation": {
    "billTo": {
      "firstName": "John",
      "lastName": "Doe",
      "address1": "201 S. Division St.",
      "postalCode": "48104-2201",
      "locality": "Ann Arbor",
      "administrativeArea": "MI",
      "country": "US",
      "phoneNumber": "5554327113",
      "email": "test@cybs.com"
    },
    "amountDetails": {
      "totalAmount": "100.00",
      "currency": "USD"
    }
```

```
  },
  "paymentInformation": {
    "card": {
      "expirationYear": "2031",
      "number": "5555xxxxxxxxxxx",
      "expirationMonth": "12"
    }
  }
}
```

Response to a Successful Request

```
{
  "_links": {
    "authReversal": {
      "method": "POST",
      "href": "/pts/v2/payments/6530824710046809304002/reversals"
    },
    "self": {
      "method": "GET",
      "href": "/pts/v2/payments/653082471004680930400 2"
    },
    "capture": {
      "method": "POST",
      "href": "/pts/v2/payments/6530824710046809304002/captures"
    }
  },
  "clientReferenceInformation": {
    "code": "1653082470983"
  },
  "id": "6530824710046809304002",
  "orderInformation": {
    "amountDetails": {
      "authorizedAmount": "100.00",
      "currency": "USD"
    }
  },
  "paymentAccountInformation": {
    "card": {
      "type": "002"
    }
  },
  "paymentInformation": {
    "tokenizedCard": {
      "type": "002"
    },
    "card": {
      "type": "002"
    }
  },
  "pointOfSaleInformation": {
    "terminalId": "111111"
  },
  "processorInformation": {
    "approvalCode": "888888",
    "authIndicator": "1",
```

```
      "networkTransactionId": "123456789619999",
      "transactionId": "123456789619999",
      "responseCode": "100",
      "avs": {
        "code": "X",
        "codeRaw": "I1"
      }
    },
    "reconciliationId": "79710341A39WTT5W",
    "status": "AUTHORIZED",
    "submitTimeUtc": "2022-05-20T21:34:31Z"
}
```


Mastercard Initial CIT Standing Order Payment with TMS

The first transaction in a standing order payment is a customer-initiated transaction (CIT). Before you can perform a subsequent merchant-initiated transaction (MIT), you must store the customer's credentials for later use. Before you can store the user's credentials, you must get the customer's consent to store their private information. This process is also known as establishing a relationship with the customer.

Creating a TMS Token

When sending the initial CIT, you can create a TMS token to store the customer's credentials for the subsequent MITs. To create a TMS token, include the **processingInformation.actionTokenTypes** field in the authorization request. Set the field to one of these values based on the TMS token type you want to create:

Customer	**Customer tokens store one or more customer payment instrument tokens and shipping address tokens.** **Including a customer token in subsequent MITs eliminates the need to include billing information, card information, and the previous transaction's ID.** `"processingInformation": { "actionTokenTypes": [ "customer" ]` **For more information about this TMS token type, see *Customer Tokens* in the Token Management Service Developer Guide.**
Payment Instrument	**Payment instrument tokens store an instrument identifier token, card information, and billing information. Payment instruments are not linked to a customer token. Including a payment instrument in subsequent MITs eliminates**

	the need to include billing information, card information, and the previous transaction's ID. `"processingInformation": { "actionTokenTypes": [ "paymentInstrument" ]` **For more information about this TMS token type, see *Payment Instrument Token* in the Token Management Service Developer Guide.**
Instrument Identifier	**Instrument identifier tokens store a PAN. Including an instrument identifier in subsequent MITs eliminates the need to include a PAN and the previous transaction's ID.** `"processingInformation": { "actionTokenTypes": [ "instrumentIdentifier" ]` **For more information about this TMS token type, see *Instrument Identifier Token* in the Token Management Service Developer Guide.**
Instrument Identifier, Payment Instrument, and Customer Identifier	**You can also create multiple TMS token types in the same authorization. This example includes an instrument identifier, a payment instrument, and a customer token in the same authorization:** `"processingInformation": { "actionTokenTypes": [ "instrumentIdentifier", "paymentInstrument", "customer" ]`

Endpoint

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

Required Fields for Authorizing Initial CIT Standing Order Payments with TMS

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.phoneNumber	
orderInformation.billTo.postalCode	
paymentInformation.card.expirationMonth	
paymentInformation.card.expirationYear	
paymentInformation.card.number	
processingInformation.actionList	**Set the value to** `TOKEN_CREATE`
processingInformation.actionTokenTypes	**Set to one or more of these values:** • `customer` • `instrumentIdentifier` • `paymentInstrument`
processingInformation.authorizationOptions.initiator.merchantInitiatedTransaction.reason	**Set the value to** `8`.
processingInformation.commerceIndicator	**Set the value to** `internet`.

REST Example: Authorizing Initial CIT Standing Order Payments with TMS

Endpoint:

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

Request

```
{
  "processingInformation": {
    "actionList": ["TOKEN_CREATE"],
    "actionTokenTypes": ["customer"],
    "commerceIndicator": "internet",
    "authorizationOptions": {
      "initiator": {
        "merchantInitiatedTransaction": {
          "reason": "8"
```

```
      }
     }
    }
   },
   "paymentInformation": {
    "card": {
     "number": "5555555555554444",
     "expirationMonth": "12",
     "expirationYear": "2031"
    }
   },
   "orderInformation": {
    "amountDetails": {
     "totalAmount": "100.00",
     "currency": "USD"
    },
    "billTo": {
     "firstName": "John",
     "lastName": "Smith",
     "address1": "123 Happy St",
     "locality": "Sunnyville",
     "administrativeArea": "CA",
     "postalCode": "55555",
     "country": "US",
     "email": "test@cybs.com",
     "phoneNumber": "444-4444-4444"
    }
   }
}
```

Response to a Successful Request

```
{
  "_links": {
    "authReversal": {
      "method": "POST",
      "href": "/pts/v2/payments/706495941148670650 3954/reversals"
    },
    "self": {
      "method": "GET",
      "href": "/pts/v2/payments/70649594114867065 03954"
    },
    "capture": {
      "method": "POST",
      "href": "/pts/v2/payments/706495941148670650 3954/captures"
    }
  },
  "clientReferenceInformation": {
    "code": "1706495941197"
  },
  "id": "706495941148670650 3954",
  "orderInformation": {
    "amountDetails": {
      "authorizedAmount": "100.00",
      "currency": "USD"
    }
```

```
  },
  "paymentAccountInformation": {
    "card": {
      "type": "002"
    }
  },
  "paymentInformation": {
    "tokenizedCard": {
      "type": "002"
    },
    "card": {
      "type": "002"
    }
  },
  "pointOfSaleInformation": {
    "terminalId": "111111"
  },
  "processorInformation": {
    "approvalCode": "888888",
    "authIndicator": "1",
    "networkTransactionId": "123456789619999",
    "transactionId": "123456789619999",
    "responseCode": "100",
    "avs": {
      "code": "X",
      "codeRaw": "I1"
    }
  },
  "reconciliationId": "680915409RRMGL34",
  "status": "AUTHORIZED",
  "submitTimeUtc": "2024-01-29T02:39:01Z",
  "tokenInformation": {
    "customer": {
      "id": "100D6CDA178DD64DE063A2598D0AD3D5"
    }
  }
}
```


Mastercard Subscription Payments

A subscription payment is a recurring COF transaction that is processed at a fixed amount at regular intervals not to exceed one year between transactions. The series of recurring payments is the result of an agreement between you and the customer for the purchase of goods or services that are provided at regular intervals.

Mastercard CIT Initial Subscription Payment

The first transaction in a subscription payment is a customer-initiated transaction (CIT). Before you can perform a subsequent merchant-initiated transaction (MIT), you must store the customer's credentials for later use. Before you can store the user's

credentials, you must get the customer's consent to store their private information. This process is also known as establishing a relationship with the customer.

Endpoint

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

Successful Response

Store the **processorInformation.networkTransactionId** field value from the successful response message. You must include the network transaction ID in subsequent MIT authorization requests to associate the CIT to the MIT.

Required Fields for Authorizing CIT Initial Subscription Payments

Use these required fields to authorize customer-initiated initial subscription payments.

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.phoneNumber	
orderInformation.billTo.postalCode	
paymentInformation.card.expirationMonth	
paymentInformation.card.expirationYear	
paymentInformation.card.number	
processingInformation.authorizationOptions.initiator.storedOnFile	**Set the value to** `true`.
processingInformation.authorizationOptions.initiator.type	**Set the value to** `customer` **or a payer authentication value.**
processingInformation.commerceIndicator	**Set the value to** `recurring`.
processingInformation.authorizationOptions.initiator.merchantInitiatedTransaction.reason	**Set the value to** `7`.

REST Example: Authorizing Initial CIT Subscription Payments

Endpoint:

- Production: POST https://api.cybersource.com/pts/v2/payments
- Test: POST https://apitest.cybersource.com/pts/v2/payments

Request

```
{
  "processingInformation": {
    "commerceIndicator": "internet",
    "authorizationOptions": {
      "initiator": {
        "type": "customer",
        "credentialStoredOnFile": "true",
        "merchantInitiatedTransaction": {
          "reason": "7"
        }
      }
    }
  },
  "orderInformation": {
    "billTo": {
      "firstName": "John",
      "lastName": "Doe",
      "address1": "201 S. Division St.",
      "postalCode": "48104-2201",
      "locality": "Ann Arbor",
      "administrativeArea": "MI",
      "country": "US",
      "phoneNumber": "5554327113",
      "email": "test@cybs.com"
    },
    "amountDetails": {
      "totalAmount": "100.00",
      "currency": "USD"
    }
  },
  "paymentInformation": {
    "card": {
      "expirationYear": "2031",
      "number": "4111xxxxxxxxxxx",
      "expirationMonth": "12"
    }
  }
}
```

Response to a Successful Request

```
{
  "_links": {
    "authReversal": {
      "method": "POST",
      "href": "/pts/v2/payments/6530824710046809304002/reversals"
    },
    "self": {
      "method": "GET",
      "href": "/pts/v2/payments/6530824710046809304002"
    },
```

```
      "capture": {
        "method": "POST",
        "href": "/pts/v2/payments/6530824710046809304002/captures"
      }
    },
    "clientReferenceInformation": {
      "code": "1653082470983"
    },
    "id": "6530824710046809304002",
    "orderInformation": {
      "amountDetails": {
        "authorizedAmount": "100.00",
        "currency": "USD"
      }
    },
    "paymentAccountInformation": {
      "card": {
        "type": "002"
      }
    },
    "paymentInformation": {
      "tokenizedCard": {
        "type": "002"
      },
      "card": {
        "type": "002"
      }
    },
    "pointOfSaleInformation": {
      "terminalId": "111111"
    },
    "processorInformation": {
      "approvalCode": "888888",
      "authIndicator": "1",
      "networkTransactionId": "123456789619999",
      "transactionId": "123456789619999",
      "responseCode": "100",
      "avs": {
        "code": "X",
        "codeRaw": "I1"
      }
    },
    "reconciliationId": "79710341A39WTT5W",
    "status": "AUTHORIZED",
    "submitTimeUtc": "2022-05-20T21:34:31Z"
}
```


Mastercard CIT Initial Subscription Payment with TMS

The first transaction in a subscription payment is a customer-initiated transaction (CIT). Before you can perform a subsequent merchant-initiated transaction (MIT), you must store the customer's credentials for later use. Before you can store the user's credentials, you must get the customer's consent to store their private information. This process is also known as establishing a relationship with the customer.

Creating a TMS Token

When sending the initial CIT, you can create a TMS token to store the customer's credentials for the subsequent MITs. To create a TMS token, include the **processingInformation.actionTokenTypes** field in the authorization request. Set the field to one of these values based on the TMS token type you want to create:

Customer

Customer tokens store one or more customer payment instrument tokens and shipping address tokens.

Including a customer token in subsequent MITs eliminates the need to include billing information, card information, and the previous transaction's ID.

```
"processingInformation": {
  "actionTokenTypes": [
   "customer"
]
```

For more information about this TMS token type, see *Customer Tokens* in the Token Management Service Developer Guide.

Payment Instrument

Payment instrument tokens store an instrument identifier token, card information, and billing information. Payment instruments are not linked to a customer token. Including a payment instrument in subsequent MITs eliminates the need to include billing information, card information, and the previous transaction's ID.

```
"processingInformation": {
  "actionTokenTypes": [
   "paymentInstrument"
]
```

For more information about this TMS token type, see *Payment Instrument Token* in the Token Management Service Developer Guide.

Instrument Identifier

Instrument identifier tokens store a PAN. Including an instrument identifier in subsequent MITs eliminates the need to include a PAN and the previous transaction's ID.

```
"processingInformation": {
```

```
  "actionTokenTypes": [
   "instrumentIdentifier"
]
```

For more information about this TMS token type, see *Instrument Identifier Token* in the Token Management Service Developer Guide.

Instrument Identifier, Payment Instrument, and Customer Identifier

You can also create multiple TMS token types in the same authorization. This example includes an instrument identifier, a payment instrument, and a customer token in the same authorization:

```
"processingInformation": {
  "actionTokenTypes": [
     "instrumentIdentifier",
     "paymentInstrument",
     "customer"
]
```

Endpoint

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

Required Fields for Authorizing CIT Initial Subscription Payments with TMS

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.phoneNumber

orderInformation.billTo.postalCode

paymentInformation.card.expirationMonth

paymentInformation.card.expirationYear

paymentInformation.card.number

processingInformation.actionList	Set the value to `TOKEN_CREATE`
processingInformation.actionTokenTypes	Set to one or more of these values: • `customer` • `instrumentIdentifier` • `paymentInstrument`
processingInformation.commerceIndicator	Set the value to `recurring`.
processingInformation.authorizationOptions.initiator.merchantInitiatedTransaction.reason	Set the value to `7`.

REST Example: Authorizing Initial CIT Subscription Payments

Endpoint:

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

Request

```
{
 "processingInformation": {
  "actionList": ["TOKEN_CREATE"],
  "actionTokenTypes": ["customer"],
  "commerceIndicator": "recurring",
  "authorizationOptions": {
   "initiator": {
    "merchantInitiatedTransaction": {
     "reason": "7"
    }
   }
  }
 },
 "paymentInformation": {
  "card": {
   "number": "5555555555554444",
   "expirationMonth": "12",
   "expirationYear": "2031"
  }
 },
 "orderInformation": {
  "amountDetails": {
   "totalAmount": "100.00",
   "currency": "USD"
  },
  "billTo": {
   "firstName": "John",
   "lastName": "Smith",
   "address1": "123 Happy St",
   "locality": "Sunnyville",
   "administrativeArea": "CA",
   "postalCode": "55555",
   "country": "US",
   "email": "test@cybs.com",
```

```
    "phoneNumber": "444-4444-4444"
   }
  }
}
```

Response to a Successful Request

```
{
 "_links": {
  "authReversal": {
   "method": "POST",
   "href": "/pts/v2/payments/706494684625641010 3954/reversals"
  },
  "self": {
   "method": "GET",
   "href": "/pts/v2/payments/70649468462564101 03954"
  },
  "capture": {
   "method": "POST",
   "href": "/pts/v2/payments/706494684625641010 3954/captures"
  }
 },
 "clientReferenceInformation": {
  "code": "1706494684667"
 },
 "id": "7064946846256410103954",
 "orderInformation": {
  "amountDetails": {
   "authorizedAmount": "100.00",
   "currency": "USD"
  }
 },
 "paymentAccountInformation": {
  "card": {
   "type": "002"
  }
 },
 "paymentInformation": {
  "tokenizedCard": {
   "type": "002"
  },
  "card": {
   "type": "002"
  }
 },
 "pointOfSaleInformation": {
  "terminalId": "111111"
 },
 "processorInformation": {
  "approvalCode": "888888",
  "authIndicator": "1",
  "networkTransactionId": "123456789619999",
  "transactionId": "123456789619999",
  "responseCode": "100",
  "avs": {
   "code": "X",
```

```
    "codeRaw": "I1"
   }
  },
  "reconciliationId": "68091233JRRDUQ34",
  "status": "AUTHORIZED",
  "submitTimeUtc": "2024-01-29T02:18:04Z",
  "tokenInformation": {
   "customer": {
    "id": "100D1DC40CC7C803E063A2598D0A29BD"
   }
  }
}
```


Unscheduled COF Payments

An unscheduled credentials-on-file (COF) transaction uses stored payment information for a fixed or variable amount that does not occur regularly. An account top-up is one kind of unscheduled COF.

Customer-Initiated Unscheduled COF Payment with PAN

An unscheduled credentials-on-file (COF) transaction uses stored payment information for a fixed or variable amount that does not occur regularly. An account top-up is one kind of unscheduled COF.

Prerequisites

The first transaction in an unscheduled COF payment is a customer-initiated transaction (CIT). Before you can perform a subsequent merchant-initiated transaction (MIT), you must store the customer's credentials for later use. Before you can store the user's credentials, you must get the customer's consent to store their private information. This process is also known as establishing a relationship with the customer.

Endpoint

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

Successful Response

Store the **processorInformation.networkTransactionId** field value from the successful response message. You must include the network transaction ID in subsequent MIT authorization requests to associate the CIT to the MIT.

Required Fields for Authorizing Initial CIT Unscheduled COF Payments

These fields are required in a subsequent authorization request for an initial unscheduled COF payment:

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.phoneNumber	
orderInformation.billTo.postalCode	
paymentInformation.card.expirationMonth	
paymentInformation.card.expirationYear	
paymentInformation.card.number	
processingInformation. authorizationOptions. initiator. credentialStoredOnFile	**Set the value to `true`.**
processingInformation. authorizationOptions. initiator. type	**Set the value to `customer`.**
processingInformation. commerceIndicator	**Set the value to `internet` or a payer authentication value.**

REST Example: Authorizing Initial CIT Unscheduled COF Payments

Endpoint:

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

Request

```
{
  "processingInformation": {
    "commerceIndicator": "internet",
    "authorizationOptions": {
      "initiator": {
        "credentialStoredOnFile": "true",
        "type": "customer"
      }
    }
  },
  "orderInformation": {
    "billTo": {
      "firstName": "John",
      "lastName": "Doe",
```

```
        "address1": "201 S. Division St.",
        "postalCode": "48104-2201",
        "locality": "Ann Arbor",
        "administrativeArea": "MI",
        "country": "US",
        "phoneNumber": "5554327113",
        "email": "test@cybs.com"
      },
      "amountDetails": {
        "totalAmount": "100.00",
        "currency": "USD"
      }
    },
    "paymentInformation": {
      "card": {
        "expirationYear": "2031",
        "number": "4111xxxxxxxxxxx",
        "expirationMonth": "12"
      }
    }
}
```

Response to a Successful Request

```
{
    "_links": {
      "authReversal": {
        "method": "POST",
        "href": "/pts/v2/payments/6528187198946076303004/reversals"
      },
      "self": {
        "method": "GET",
        "href": "/pts/v2/payments/652818719894607630304"
      },
      "capture": {
        "method": "POST",
        "href": "/pts/v2/payments/6528187198946076303004/captures"
      }
    },
    "clientReferenceInformation": {
      "code": "1652818719876"
    },
    "id": "652818719894607630304",
    "orderInformation": {
      "amountDetails": {
        "authorizedAmount": "100.00",
        "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": "63165088Z3AHV91G",
    "status": "AUTHORIZED",
    "submitTimeUtc": "2022-05-17T20:18:40Z"
}
```


Customer-Initiated Unscheduled COF Payments with TMS

An unscheduled credentials-on-file (COF) transaction uses stored payment information for a fixed or variable amount that does not occur regularly. An account top-up is one kind of unscheduled COF.

Prerequisites

The first transaction in an unscheduled COF payment is a customer-initiated transaction (CIT). Before you can perform a subsequent merchant-initiated transaction (MIT), you must store the customer's credentials for later use. Before you can store the user's credentials, you must get the customer's consent to store their private information. This process is also known as establishing a relationship with the customer.

Creating a TMS Token

When sending the initial CIT, you can create a TMS token to store the customer's credentials for the subsequent MITs. To create a TMS token, include the **processingInformation.actionTokenTypes** field in the authorization request. Set the field to one of these values based on the TMS token type you want to create:

Customer	**Customer tokens store one or more customer payment instrument tokens and shipping address tokens.** **Including a customer token in subsequent MITs eliminates the need to include billing**

	information, card information, and the previous transaction's ID. `"processingInformation": { "actionTokenTypes": [ "customer" ]` For more information about this TMS token type, see *Customer Tokens* in the Token Management Service Developer Guide.
Payment Instrument	Payment instrument tokens store an instrument identifier token, card information, and billing information. Payment instruments are not linked to a customer token. Including a payment instrument in subsequent MITs eliminates the need to include billing information, card information, and the previous transaction's ID. `"processingInformation": { "actionTokenTypes": [ "paymentInstrument" ]` For more information about this TMS token type, see *Payment Instrument Token* in the Token Management Service Developer Guide.
Instrument Identifier	Instrument identifier tokens store a PAN. Including an instrument identifier in subsequent MITs eliminates the need to include a PAN and the previous transaction's ID. `"processingInformation": { "actionTokenTypes": [ "instrumentIdentifier" ]` For more information about this TMS token type, see *Instrument Identifier Token* in the Token Management Service Developer Guide.
Instrument Identifier, Payment Instrument, and Customer Identifier	You can also create multiple TMS token types in the same authorization. This example includes an instrument identifier, a

payment instrument, and a customer token in the same authorization:

```
"processingInformation": {
  "actionTokenTypes": [
    "instrumentIdentifier",
    "paymentInstrument",
    "customer"
]
```

Endpoint

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

Required Fields for CIT Unscheduled COF Payments with TMS

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.phoneNumber	
orderInformation.billTo.postalCode	
paymentInformation.card.expirationMonth	
paymentInformation.card.expirationYear	
paymentInformation.card.number	
processingInformation.actionList	Set the value to `TOKEN_CREATE`
processingInformation.actionTokenTypes	Set to one or more of these values: • `customer` • `instrumentIdentifier` • `paymnentInstrument`
processingInformation.commerceIndicator	Set the value to `internet` or a payer authentication value.

REST Example: Initial CIT Unscheduled COF Payment in TMS

Endpoint:

- Production: POST https://api.cybersource.com/pts/v2/payments
- Test: POST https://apitest.cybersource.com/pts/v2/payments

Request

```
{
  "processingInformation": {
    "actionList": [
      "TOKEN_CREATE"
    ],
    "actionTokenTypes": [
      "customer"
    ],
    "commerceIndicator": "internet"
  },
  "paymentInformation": {
    "card": {
      "number": "4111111111111111",
      "expirationMonth": "12",
      "expirationYear": "2031"
    }
  },
  "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": "444-4444-4444"
    }
  }
}
```

Response to a Successful Request

```
{
  "_links": {
    "authReversal": {
      "method": "POST",
      "href": "/pts/v2/payments/6976866073586557303955/reversals"
    },
    "self": {
      "method": "GET",
      "href": "/pts/v2/payments/6976866073586557303955"
```

```
    },
    "capture": {
      "method": "POST",
      "href": "/pts/v2/payments/6976866073586557303955/captures"
    }
  },
  "clientReferenceInformation": {
    "code": "1697686607441"
  },
  "id": "6976866073586557303955",
  "orderInformation": {
    "amountDetails": {
      "authorizedAmount": "102.21",
      "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": "62699023FNN143DG",
  "status": "AUTHORIZED",
  "submitTimeUtc": "2023-10-19T03:36:47Z",
  "tokenInformation": {
    "customer": {
      "id": "080A6C3842C72DCBE063A2598D0AA98B"
    }
  }
}
```

Merchant-Initiated Unscheduled COF Payments with PAN

After the initial CIT unscheduled COF payment, subsequent unscheduled COF transactions are merchant-initiated transactions (MITs).

Endpoint

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

Required Fields for Authorizing Subsequent MIT Unscheduled COF Payments

These fields are required in a subsequent authorization request for a subsequent unscheduled COF payment:

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.phoneNumber	
orderInformation.billTo.postalCode	
paymentInformation.card.expirationMonth	
paymentInformation.card.expirationYear	
paymentInformation.card.number	
processingInformation. authorizationOptions. initiator. merchantInitiatedTransaction. previousTransactionID	**For Discover, Mastercard, and American Express cards, use the transaction ID from the original transaction.** For Visa, use the last successful transaction ID.
processingInformation. authorizationOptions. initiator. storedCredentialUsed	**Set the value to `true`.**
processingInformation. authorizationOptions. initiator. type	**Set the value to `merchant`.**
processingInformation. commerceIndicator	**Set the value to `internet`.**

Card-Specific Required Field for Processing a Merchant-Initiated Transactions

Discover

The listed card requires an additional field:

processingInformation. authorizationOptions. initiator. merchantInitiatedTransaction. originalAuthorizedAmount	**Provide the original transaction amount.**

Country-Specific Required Fields for Authorizing Subsequent Recurring Payments

Include these country-specific required fields for a successful merchant-initiated authorization.

India

These fields are required only with Diners Club in India or with an India-issued card, and you are processing payments through Visa Platform Connect.

installmentInformation.amount

installmentInformation.frequency

installmentInformation.identifier

installmentInformation.paymentType

installmentInformation.sequence

installmentInformation.validationIndicator

Saudi Arabia

These fields are required only if your business is located in Saudi Arabia and you are processing payments through Visa Platform Connect.

authorizationOptions.initiator.merchantInitiatedTransaction.agreementId

recurringPaymentInformation.amountType

REST Example: Authorizing Subsequent MIT Unscheduled COF Payments

Endpoint:

- Production: POST https://api.cybersource.com/pts/v2/payments
- Test: POST https://apitest.cybersource.com/pts/v2/payments

Request

```
{
  "processingInformation": {
    "commerceIndicator": "internet",
    "authorizationOptions": {
      "initiator": {
        "storedCredentialUsed": "true",
```

```
          "type": "merchant",
          "merchantInitiatedTransaction": {
            "previousTransactionId": "123456789619999",
            "originalAuthorizedAmount": "100"    <--Discover Only-->
          }
        }
      }
    },
    "orderInformation": {
      "billTo": {
        "firstName": "John",
        "lastName": "Doe",
        "address1": "201 S. Division St.",
        "postalCode": "48104-2201",
        "locality": "Ann Arbor",
        "administrativeArea": "MI",
        "country": "US",
        "phoneNumber": "5554327113",
        "email": "test@cybs.com"
      },
      "amountDetails": {
        "totalAmount": "100.00",
        "currency": "USD"
      }
    },
    "paymentInformation": {
      "card": {
        "expirationYear": "2031",
        "number": "4111xxxxxxxxxxx",
        "expirationMonth": "12"
      }
    }
}
```

Response to a Successful Request

```
{
    "_links": {
      "authReversal": {
        "method": "POST",
        "href": "/pts/v2/payments/6530824710046809304002/reversals"
      },
      "self": {
        "method": "GET",
        "href": "/pts/v2/payments/653082471004680930402"
      },
      "capture": {
        "method": "POST",
        "href": "/pts/v2/payments/6530824710046809304002/captures"
      }
    },
    "clientReferenceInformation": {
      "code": "1653082470983"
    },
    "id": "6530824710046809304002",
    "orderInformation": {
```

```
      "amountDetails": {
        "authorizedAmount": "100.00",
        "currency": "USD"
      }
    },
    "paymentAccountInformation": {
      "card": {
        "type": "002"
      }
    },
    "paymentInformation": {
      "tokenizedCard": {
        "type": "002"
      },
      "card": {
        "type": "002"
      }
    },
    "pointOfSaleInformation": {
      "terminalId": "111111"
    },
    "processorInformation": {
      "approvalCode": "888888",
      "authIndicator": "1",
      "networkTransactionId": "123456789619999",
      "transactionId": "123456789619999",
      "responseCode": "100",
      "avs": {
        "code": "X",
        "codeRaw": "I1"
      }
    },
    "reconciliationId": "79710341A39WTT5W",
    "status": "AUTHORIZED",
    "submitTimeUtc": "2022-05-20T21:34:31Z"
}
```


Merchant-Initiated Unscheduled COF Payments with TMS

After the customer-initiated unscheduled COF payment, you can send merchant-initiated unscheduled COF payments using one or more TMS token types:

Customer	Customer tokens store one or more customer payment instrument tokens and shipping address tokens. Including a customer token eliminates the need to include billing information, card information, and the previous transaction's ID.


```
"paymentInformation": {
 "customer": {
  "id": "07C9CA98022DA498E063A2598D0AA400"
 }
```

	<pre>}</pre> For more information about this TMS token type, see *Customer Tokens* in the Token Management Service Developer Guide.

Payment Instrument	Payment instrument tokens store an instrument identifier token, card information, and billing information. Payment instruments are not linked to a customer token.
	Including a payment instrument eliminates the need to include billing information, card information, and the previous transaction's ID.
	<pre>"paymentInformation": {
	  "paymentInstrument": {
	  "id": "07CA24EF20F9E2C9E063A2598D0A8565"
	  }
	}</pre>
	For more information about this TMS token type, see *Payment Instrument Token* in the Token Management Service Developer Guide.

Instrument Identifier	Instrument identifier tokens store only a PAN. Including an instrument identifier eliminates the need to include a PAN and the previous transaction's ID.
	<pre>"paymentInformation": {
	  "instrumentIdentifier": {
	  "id": "7010000000016241111"
	  }
	}</pre>
	For more information about this TMS token type, see *Instrument Identifier Token* in the Token Management Service Developer Guide.

Endpoint

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

Required Fields for MIT Unscheduled COF Payments with TMS

Include these Required Fields

orderInformation.amountDetails.currency

orderInformation.amountDetails.totalAmount

paymentInformation.[tokentype].id	Where **[tokentype]** is the TMS token type you are using: • **customer** • **instrumentIdentifier** • **paymentInstrument**
processingInformation.commerceIndicator	Set the value to `internet`.

Instrument Identifier Required Fields

If you are using the **paymentInformation.instrumentIdentifier.id** token, include these required fields in addition to the required fields listed above.

orderInformation.billTo.address1

orderInformation.billTo.administrativeArea

orderInformation.billTo.country

orderInformation.billTo.email

orderInformation.billTo.firstName

orderInformation.billTo.lastName

orderInformation.billTo.locality

orderInformation.billTo.phoneNumber

orderInformation.billTo.postalCode

paymentInformation.card.expirationMonth

paymentInformation.card.expirationYear

Card-Specific Field

The listed card type requires an additional field.

Discover	**processingInformation.authorizationOptions.initiator.merchantInitiatedTransaction.originalAuthorizedAmou** **Provide the original transaction amount.**

Country-Specific Fields

Include these country-specific required fields for a successful merchant-initiated authorization.

India	**These fields are required only with Diners Club in India or with an India-issued card, and you are processing payments through Visa Platform Connect.** **installmentInformation.amount** **installmentInformation.frequency** **installmentInformation.identifier** **installmentInformation.paymentType** **installmentInformation.sequence** **installmentInformation.validationIndicator**
Saudi Arabia	**These fields are required only if your business is located in Saudi Arabia and you are processing payments through Visa Platform Connect.** **authorizationOptions.initiator.merchantInitiatedTransa** **recurringPaymentInformation.amountType**

Example: MIT Unscheduled COF Payment with TMS Instrument Identifier

Endpoint:

- Production: POST https://api.cybersource.com/pts/v2/payments
- Test: POST https://apitest.cybersource.com/pts/v2/payments

Request

```
{
 "processingInformation": {
  "commerceIndicator": "internet"
 },
 "paymentInformation": {
  "card": {
   "expirationMonth": "12",
   "expirationYear": "2031"
  },
  "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"
   }
  }
}
```

Response to a Successful Request

```
{
  "_links": {
   "authReversal": {
    "method": "POST",
    "href": "/pts/v2/payments/6976892714556134003954/reversals"
   },
   "self": {
    "method": "GET",
    "href": "/pts/v2/payments/6976892714556134003954"
   },
   "capture": {
    "method": "POST",
    "href": "/pts/v2/payments/6976892714556134003954/captures"
   }
  },
  "clientReferenceInformation": {
   "code": "1697689271513"
  },
  "id": "6976892714556134003954",
  "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"
  },
  "processingInformation": {
```

```
    "paymentSolution": "015"
   },
   "processorInformation": {
    "paymentAccountReferenceNumber": "V0010013022298169667504231315",
    "approvalCode": "888888",
    "networkTransactionId": "123456789619999",
    "transactionId": "123456789619999",
    "responseCode": "100",
    "avs": {
     "code": "X",
     "codeRaw": "I1"
    }
   },
   "reconciliationId": "62699554NNMR6X7R",
   "status": "AUTHORIZED",
   "submitTimeUtc": "2023-10-19T04:21:11Z"
}
```

Example: MIT Unscheduled COF Payment with TMS Payment Instrument

Endpoint:

- Production: POST https://api.cybersource.com/pts/v2/payments
- Test: POST https://apitest.cybersource.com/pts/v2/payments

Request

```
{
   "processingInformation": {
    "commerceIndicator": "internet"
   },
   "paymentInformation": {
    "paymentInstrument": {
     "id": "080AE120369A7947E063A2598D0A718F"
    }
   },
   "orderInformation": {
    "amountDetails": {
     "totalAmount": "102.21",
     "currency": "USD"
    }
   }
}
```

Response to a Successful Request

```
{
   "_links": {
    "authReversal": {
     "method": "POST",
     "href": "/pts/v2/payments/6976891300676431103955/reversals"
    },
    "self": {
     "method": "GET",
     "href": "/pts/v2/payments/6976891300676431103955"
    },
```

```
    "capture": {
      "method": "POST",
      "href": "/pts/v2/payments/6976891300676431103955/captures"
    }
  },
  "clientReferenceInformation": {
    "code": "1697689130124"
  },
  "id": "6976891300676431103955",
  "orderInformation": {
    "amountDetails": {
      "authorizedAmount": "102.21",
      "currency": "USD"
    }
  },
  "paymentAccountInformation": {
    "card": {
      "type": "001"
    }
  },
  "paymentInformation": {
    "tokenizedCard": {
      "type": "001"
    },
    "instrumentIdentifier": {
      "id": "7010000000016241111",
      "state": "ACTIVE"
    },
    "paymentInstrument": {
      "id": "080AE120369A7947E063A2598D0A718F"
    },
    "card": {
      "type": "001"
    }
  },
  "pointOfSaleInformation": {
    "terminalId": "111111"
  },
  "processingInformation": {
    "paymentSolution": "015"
  },
  "processorInformation": {
    "paymentAccountReferenceNumber": "V0010013022298169667504231315",
    "approvalCode": "888888",
    "networkTransactionId": "123456789619999",
    "transactionId": "123456789619999",
    "responseCode": "100",
    "avs": {
      "code": "X",
      "codeRaw": "I1"
    }
  },
  "reconciliationId": "62699372XNMR85HS",
  "status": "AUTHORIZED",
  "submitTimeUtc": "2023-10-19T04:18:50Z"
}
```

Example: MIT Unscheduled COF Payment with TMS Customer

Endpoint:

- Production: POST https://api.cybersource.com/pts/v2/payments
- Test: POST https://apitest.cybersource.com/pts/v2/payments

Request

```
{
 "processingInformation": {
  "commerceIndicator": "internet"
 },
 "paymentInformation": {
  "customer": {
   "id": "080AC9AB60C92AA2E063A2598D0A0C74"
  }
 },
 "orderInformation": {
  "amountDetails": {
   "totalAmount": "102.21",
   "currency": "USD"
  }
 }
}
```

Response to a Successful Request

```
{
 "_links": {
  "authReversal": {
   "method": "POST",
   "href": "/pts/v2/payments/6976889582016147703955/reversals"
  },
  "self": {
   "method": "GET",
   "href": "/pts/v2/payments/6976889582016147703955"
  },
  "capture": {
   "method": "POST",
   "href": "/pts/v2/payments/6976889582016147703955/captures"
  }
 },
 "clientReferenceInformation": {
  "code": "1697688958296"
 },
 "id": "6976889582016147703955",
 "orderInformation": {
  "amountDetails": {
   "authorizedAmount": "102.21",
   "currency": "USD"
  }
 },
 "paymentAccountInformation": {
  "card": {
   "type": "001"
```

```
    }
   },
   "paymentInformation": {
    "tokenizedCard": {
     "type": "001"
    },
    "instrumentIdentifier": {
     "id": "7010000000016241111",
     "state": "ACTIVE"
    },
    "paymentInstrument": {
     "id": "080AE6DB37B09557E063A2598D0AA4C9"
    },
    "card": {
     "type": "001"
    },
    "customer": {
     "id": "080AC9AB60C92AA2E063A2598D0A0C74"
    }
   },
   "pointOfSaleInformation": {
    "terminalId": "111111"
   },
   "processingInformation": {
    "paymentSolution": "015"
   },
   "processorInformation": {
    "paymentAccountReferenceNumber": "V0010013022298169667504231315",
    "approvalCode": "888888",
    "networkTransactionId": "123456789619999",
    "transactionId": "123456789619999",
    "responseCode": "100",
    "avs": {
     "code": "X",
     "codeRaw": "I1"
    }
   },
   "reconciliationId": "62699842BNN13VA0",
   "status": "AUTHORIZED",
   "submitTimeUtc": "2023-10-19T04:15:58Z"
}
```

Token Management Service Processing

This section provides the information you need in order to process Token Management Service authorization and credit transactions.

> **Important**
>
> Due to mandates from the Reserve Bank of India, Indian merchants cannot store personal account numbers (PANs). Use network tokens instead. For more information on network tokens, see the Network Tokenization section of the *Token Management Service Guide.*

Additional Resources for TMS

For more information, see these guides:

- *Token Management Service Developer Guide*
- *API field reference guide for the REST API*
- Github repositories: *https://github.com/Cybersource?q=rest-sample&type=all&language=&sort=*

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

Endpoint:

- Production: POST https://api.cybersource.com/pts/v2/payments
- Test: POST https://apitest.cybersource.com/pts/v2/payments

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": "7055928871556818104953",
  "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

Endpoint:

- Production: POST https://api.cybersource.com/pts/v2/payments
- Test: POST https://apitest.cybersource.com/pts/v2/payments

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/6778647071126384904953/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": "01103220000000E100002000....."
  },
  "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"
}
```

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

Endpoint:

- Production: POST https://api.cybersource.com/pts/v2/payments
- Test: POST https://apitest.cybersource.com/pts/v2/payments

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": "703000000000 14831523",
   "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"
}
```


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

Endpoint:

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

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/705595264858666533004951/reversals"
  },
  "self": {
   "method": "GET",
   "href": "/pts/v2/payments/70559526485866533004951"
  },
  "capture": {
   "method": "POST",
   "href": "/pts/v2/payments/705595264858666533004951/captures"
  }
 },
 "clientReferenceInformation": {
  "code": "12345678"
 },
 "id": "70559526485866533004951",
 "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"
}
```


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

Endpoint:

- Production: POST https://api.cybersource.com/pts/v2/payments
- Test: POST https://apitest.cybersource.com/pts/v2/payments

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/6765713628736138103955/reversals"
      },
      "self": {
        "method": "GET",
        "href": "/pts/v2/payments/676571362873613810 3955"
      },
      "capture": {
        "method": "POST",
        "href": "/pts/v2/payments/6765713628736138103955/captures"
      }
    },
    "clientReferenceInformation": {
      "code": "12345678"
    },
    "id": "6765713628736138103955",
    "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"
}
```


Authorizing 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 Interactive Example: Authorizing a Payment with an Instrument Identifier

Authorization with Instrument Identifier Token Id

Live Console URL: *https://developer.cybersource.com/api-reference-assets/index.html#payments_payments_process-a-payment_samplerequests-dropdown_authorization-using-tokens_authorization-with-instrument-identifier-token-id_liveconsole-tab-request-body*

REST Example: Authorizing a Payment with an Instrument Identifier

Endpoint:

- Production: POST https://api.cybersource.com/pts/v2/payments
- Test: POST https://apitest.cybersource.com/pts/v2/payments

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/7055955288186053404953/reversals"
    },
    "self": {
      "method": "GET",
      "href": "/pts/v2/payments/705595528818605340 4953"
    },
    "capture": {
      "method": "POST",
```

```
      "href": "/pts/v2/payments/705595528818605340 4953/captures"
    }
  },
  "clientReferenceInformation": {
    "code": "12345678"
  },
  "id": "7055955288186053404953",
  "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

This example shows you how to authorize a payment using an instrument identifier token while creating customer, payment instrument, and shipping address tokens.

Endpoint:

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

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

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/711467984037668720 3955/reversals"
   },
   "self": {
    "method": "GET",
    "href": "/pts/v2/payments/71146798403766872 03955"
   },
   "capture": {
    "method": "POST",
    "href": "/pts/v2/payments/711467984037668720 3955/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"
    }
  }
}
```

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 Using the REST API

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

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

{
  "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"
  }
}
```

Successful Response

```
{
  "_links": {
    "authReversal": {
      "method": "POST",
      "href": "/pts/v2/payments/6769913443166412604951/reversals"
    },
    "self": {
      "method": "GET",
      "href": "/pts/v2/payments/676991344316641260 4951"
    },
    "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"
}
```

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 Interactive Example: Authorizing a Payment with a Legacy Token

Authorization with Legacy Token

Live Console URL: *https://developer.cybersource.com/api-reference-assets/index.html#payments_payments_process-a-payment_samplerequests-dropdown_authorization-using-tokens_authorization-with-legacy-token_liveconsole-tab-request-body*

REST Example: Authorizing 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

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/705595634247678900 4951/reversals"
    },
    "self": {
      "method": "GET",
      "href": "/pts/v2/payments/70559563424767890 04951"
    },
    "capture": {
      "method": "POST",
      "href": "/pts/v2/payments/705595634247678900 4951/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 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

Endpoint:

- Production: POST https://api.cybersource.com/pts/v2/credits/
- Test: POST https://apitest.cybersource.com/pts/v2/credits/

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/705596767782613290 4951/voids"
    },
    "self": {
      "method": "GET",
      "href": "/pts/v2/credits/70559676778261329 04951"
    }
  },
  "clientReferenceInformation": {
    "code": "12345678"
  },
  "creditAmountDetails": {
    "currency": "USD",
    "creditAmount": "10.00"
  },
  "id": "705596767782613290 4951",
  "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": "V001001301932612153813096266",
    "approvalCode": "888888",
    "responseCode": "100"
  },
  "reconciliationId": "67444961BRIL0BB8",
  "status": "PENDING",
  "submitTimeUtc": "2024-01-18T16:52:48Z"
}
```


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

Endpoint:

- Production: POST https://api.cybersource.com/pts/v2/credits/
- Test: POST https://apitest.cybersource.com/pts/v2/credits/

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/705596858138644610 4953/voids"
  },
  "self": {
   "method": "GET",
   "href": "/pts/v2/credits/70559685813864461 04953"
  }
 },
 "clientReferenceInformation": {
  "code": "12345678"
 },
 "creditAmountDetails": {
  "currency": "USD",
  "creditAmount": "10.00"
 },
 "id": "7055968581386446104953",
 "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"
}
```

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

Endpoint:

- Production: POST https://api.cybersource.com/pts/v2/credits/
- Test: POST https://apitest.cybersource.com/pts/v2/credits/

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/705596958668646710 4953/voids"
  },
  "self": {
   "method": "GET",
   "href": "/pts/v2/credits/7055969586686467104953"
  }
 },
 "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"
}
```

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 Interactive Example: Making a Credit with an Instrument Identifier

Credit with Instrument Identifier Token Id

Live Console URL: *https://developer.cybersource.com/api-reference-assets/index.html#payments_credit_process-a-credit_samplerequests-dropdown_credit-with-tokenization_credit-with-instrument-identifier-token-id_liveconsole-tab-request-body*

REST Example: Making a Credit with an Instrument Identifier

Endpoint:

- Production: POST https://api.cybersource.com/pts/v2/credits/
- Test: POST https://apitest.cybersource.com/pts/v2/credits/

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/705597026106621240 4951/voids"
  },
  "self": {
   "method": "GET",
   "href": "/pts/v2/credits/705597026106621240 4951"
  }
 },
 "clientReferenceInformation": {
  "code": "12345678"
 },
 "creditAmountDetails": {
  "currency": "USD",
  "creditAmount": "10.00"
 },
 "id": "7055970261066212404951",
 "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"
}
```


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

Endpoint:

- Production: POST https://api.cybersource.com/pts/v2/credits/
- Test: POST https://apitest.cybersource.com/pts/v2/credits/

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/7055970562096509704953/voids"
    },
    "self": {
      "method": "GET",
      "href": "/pts/v2/credits/705597056209650970 4953"
    }
  },
  "clientReferenceInformation": {
    "code": "12345678"
  },
  "creditAmountDetails": {
    "currency": "USD",
    "creditAmount": "22.00"
  },
  "id": "7055970562096509704953",
  "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"
}
```