

Samsung Pay

Using the SCMP API

May 2019

CyberSource[®]
the power of payment

CyberSource Contact Information

For general information about our company, products, and services, go to <http://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: <http://www.cybersource.com/support>

Copyright

© 2019 CyberSource Corporation. All rights reserved. CyberSource Corporation ("CyberSource") furnishes this document and the software described in this document under the applicable agreement between the reader of this document ("You") and CyberSource ("Agreement"). You may use this document and/or software only in accordance with the terms of the Agreement. Except as expressly set forth in the Agreement, the information contained in this document is subject to change without notice and therefore should not be interpreted in any way as a guarantee or warranty by CyberSource. CyberSource assumes no responsibility or liability for any errors that may appear in this document. The copyrighted software that accompanies this document is licensed to You for use only in strict accordance with the Agreement. You should read the Agreement carefully before using the software. Except as permitted by the Agreement, You may not reproduce any part of this document, store this document in a retrieval system, or transmit this document, in any form or by any means, electronic, mechanical, recording, or otherwise, without the prior written consent of CyberSource.

Restricted Rights Legends

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

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 Unpublished rights reserved under the copyright laws of the United States.

Trademarks

Authorize.Net, eCheck.Net, and The Power of Payment are registered trademarks of CyberSource Corporation.

CyberSource, CyberSource Payment Manager, CyberSource Risk Manager, CyberSource Decision Manager, and CyberSource Connect are trademarks and/or service marks of CyberSource Corporation.

All other brands and product names are trademarks or registered trademarks of their respective owners.

Contents

Recent Revisions to This Document 5

About This Guide 7

Audience and Purpose 7

Conventions 7

Notes and Important Statements 7

Text and Command Conventions 8

Related Documents 8

Customer Support 8

Chapter 1 Introduction 9

Requirements 9

Supported Processors, Card Types, and Optional Features 10

Transaction Endpoints 12

Chapter 2 Registration 13

Registering with Samsung 13

Registering with CyberSource 14

Chapter 3 Integrating the Samsung SDK 16

Creating a Project 16

Integrating the Samsung Pay SDK 17

Using the API Key 17

Chapter 4 Using the Samsung Pay SDK 18

- Eligibility 18
- Payment Request 19
 - Initiating a Payment 19
 - Requesting a Payment 21

Chapter 5 Authorizing a Payment 23

- Merchant Decryption 23
 - Visa Transaction 23
 - Mastercard Transaction 25
 - American Express Transaction 27
 - JCB Transaction 29
- CyberSource Decryption 31
 - Visa Transaction 31
 - Mastercard Transaction 33
 - American Express Transaction 35
 - JCB Transaction 37
- Additional CyberSource Services 38

Appendix A API Fields 39

- Data Type Definitions 39
- Relaxed Requirements for Address Data 39
- API Request Fields 40
- Offer-Level Fields 47
- API Reply Fields 47

Recent Revisions to This Document

| Release | Changes |
|---------------|--|
| May 2019 | This revision contains only editorial changes and no technical updates. |
| April 2019 | <p>Added support for tokenized transactions using a network token with 3D Secure or SecureCode. See "Merchant Decryption," page 23.</p> <p>Added the following request fields that support tokenized transactions using a network token with 3D Secure or SecureCode (see "API Request Fields," page 40):</p> <ul style="list-style-type: none"> ■ <code>directory_server_transaction_id</code> ■ <code>network_token_cryptogram</code> ■ <code>pa_specification_version</code> <p>Added the following reply field that supports tokenized transactions using a network token with 3D Secure or SecureCode (see "API Reply Fields," page 47):</p> <p><code>directory_server_transaction_id</code></p> <p>Added support for the processor <i>Elavon Americas</i>. See "Supported Processors, Card Types, and Optional Features," page 10.</p> <p>Added support for the following optional features by Elavon Americas (see "Supported Processors, Card Types, and Optional Features," page 10):</p> <ul style="list-style-type: none"> ■ Merchant-Initiated transactions ■ Multiple partial captures ■ Recurring payments |
| February 2019 | Updated URLs for Samsung Pay Partner Portal. See "Registering with Samsung," page 13 . |

| Release | Changes |
|---------------|---|
| January 2019 | <p>Updated "JCB Transaction," page 29 and "JCB Transaction," page 29 to correct an erroneous Product Service code, Authorization Service fieldname, and payment descriptor.</p> <p>Updated "Supported Processors, Card Types, and Optional Features," page 10 to remove erroneous content regarding Vantiv.</p> <p>Updated URLs for the following:</p> <ul style="list-style-type: none"> ■ Samsung Pay Partner Portal (see "Related Documents," page 8) ■ Transaction Endpoints (see "Transaction Endpoints," page 12) ■ Samsung Pay registration (see "Registering with Samsung," page 13) ■ Decrypting payment credentials (see "Encrypted Payment Credential," page 22) |
| July 2018 | <p>All processors: updated information about optional features. See "Supported Processors, Card Types, and Optional Features," page 10.</p> |
| December 2017 | <p>Added the following to the list of supported processors. See "Supported Processors, Card Types, and Optional Features," page 10:</p> <ul style="list-style-type: none"> ■ JCN Gateway ■ Barclays ■ GPN ■ OmniPay Direct ■ Streamline |

About This Guide

Audience and Purpose

This document is written for merchants who want to enable customers to use Samsung Pay to pay for in-app purchases. This document provides an overview of integrating the Samsung Pay SDK and describes how to request the CyberSource API to process an authorization.

This document describes the Samsung Pay SDK and the CyberSource API. See "[Using the Samsung Pay SDK](#)," page 18, and "[Authorizing a Payment](#)," page 23. Merchants must use the Samsung Pay SDK to receive the customer's encrypted payment data before requesting the CyberSource API to process the transaction.

Conventions

Notes and Important Statements



Note

A *Note* contains helpful suggestions or references to material not contained in the document.



Important

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

Text and Command Conventions

| Convention | Usage |
|-------------|--|
| bold | <ul style="list-style-type: none"> Field and service names in text; for example: Include the customer_cc_number field. Items that you are instructed to act upon; for example: Click Save. |
| Screen text | Code examples and samples. |

Related Documents

CyberSource Documents:

- [Getting Started with CyberSource Advanced for the SCMP API \(PDF | HTML\)](#)
- [SCMP API Documentation and Downloads page](#)
- [Credit Card Services Using the SCMP API \(PDF | HTML\)](#)
- [Payment Network Tokenization Using the SCMP API \(PDF | HTML\)](#)

Samsung Pay documents:

[Samsung Pay Partner Portal](#)

Refer to the Support Center for complete CyberSource technical documentation:

http://www.cybersource.com/support_center/support_documentation

Customer Support

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

<http://www.cybersource.com/support>

Introduction

Requirements



ProductName relies on payment network tokenization. You can sign up for ProductName only if both of the following statements are true:

- Your processor supports payment network tokenization.
- CyberSource supports payment network tokenization with your processor.

If one or both of the preceding statements are not true, you must take one of the following actions before you can sign up for ProductName:

- Obtain a new merchant account with a processor that supports payment network tokenization.
 - Wait until your processor supports payment network tokenization.
-

You must create:

- A CyberSource account. If you do not already have a CyberSource account, contact your local CyberSource sales representative:
<http://www.cybersource.com/locations/>
 - A merchant account with a supported processor. See "[Supported Processors, Card Types, and Optional Features](#)," page 10.
 - A profile on the Samsung Pay Partner Portal, and you must obtain a Partner ID. See "[Registration](#)," page 13.
-



All optional features are described in [Payment Network Tokenization Using the SCMP API](#).

Supported Processors, Card Types, and Optional Features



Note

All optional features, except split shipments, are described in *Payment Network Tokenization Using the SCMP API* ([PDF](#) | [HTML](#)). Split shipments are described in *Credit Card Services Using the SCMP API* ([PDF](#) | [HTML](#)).

Table 1 Supported Processors, Card Types, and Optional Features

| Processors | Card Types | Optional |
|--|---|--|
| American Express Direct | American Express | <ul style="list-style-type: none"> ■ Multiple partial captures ■ Recurring Payments |
| Barclays | Visa, Mastercard, JCB, Maestro (International), Maestro (UK Domestic) If you support Maestro (UK Domestic), you must also support Maestro (International), and you must support Mastercard SecureCode for both card types. | <ul style="list-style-type: none"> ■ Multiple partial captures ■ Recurring Payments |
| Chase Paymentech Solutions | Visa, Mastercard, American Express, Discover, Diners Club, JCB, Carte Blanche, Maestro (International) | <ul style="list-style-type: none"> ■ Multiple partial captures ■ Recurring Payments |
| CyberSource through VisaNet. The supported acquirer is: <ul style="list-style-type: none"> ■ Vantiv CyberSource through VisaNet is a single processor with multiple acquirers. | Visa, Mastercard, American Express, Discover, JCB, Diners Club | <ul style="list-style-type: none"> ■ Split shipments. ■ Recurring Payments. |
| Elavon Americas | Visa, Mastercard, American Express, JCB, Diners Club, Discover, China UnionPay | <ul style="list-style-type: none"> ■ Merchant-Initiated transactions ■ Multiple partial captures ■ Recurring payments |
| FDC Compass | Visa, Mastercard, American Express, Discover, Diners Club, JCB | <ul style="list-style-type: none"> ■ Multiple partial captures ■ Recurring Payments |

Table 1 Supported Processors, Card Types, and Optional Features (Continued)

| Processors | Card Types | Optional |
|---|--|--|
| FDC Nashville Global | Visa, Mastercard, American Express, Discover, Diners Club, JCB, China UnionPay | <ul style="list-style-type: none"> ■ Multiple partial captures. ■ Recurring Payments |
| GPN | Visa, Mastercard, American Express, Discover, Diners Club, JCB | <ul style="list-style-type: none"> ■ Split shipments ■ Recurring Payments |
| JCN Gateway | Visa, Mastercard, American Express, Diners Club, JCB, NICOS house card, ORICO house card | Multiple partial captures |
| OmniPay Direct <ul style="list-style-type: none"> ■ Bank of America Merchant Services ■ First Data Merchant Solutions (Europe) ■ Global Payments International Acquiring | Visa, Mastercard, Discover, Diners Club, Maestro (UK Domestic), Maestro (International) | <ul style="list-style-type: none"> ■ Multiple partial captures ■ Recurring Payments |
| Streamline | Visa, Mastercard | <ul style="list-style-type: none"> ■ Multiple partial captures ■ Recurring Payments ■ Subsequent authorizations |
| TSYS Acquiring Solutions | Visa, Mastercard, American Express | <ul style="list-style-type: none"> ■ Multiple partial captures ■ Recurring Payments |

Transaction Endpoints

CAS (test transactions):

- Akamai endpoints:

<http://ics2testa.ic3.com>

- Non-Akamai endpoints:

<http://ics2test.ic3.com/>

Production (live transactions):

- Akamai endpoints:

<http://ics2a.ic3.com>

- Non-Akamai endpoints

<http://ics2.ic3.com/>

Registration

Registering with Samsung

To register with Samsung:

- Step 1** Create a profile by completing the merchant application on the [Samsung Pay Partner Portal](#).

**Note**

Samsung will contact you if clarifications are required.

- Step 2** After your merchant application is approved, you receive a unique partner ID. Include this ID in your application.

**Important**

You need the partner ID in order to generate the Certificate Signing Request (CSR) file in using the CyberSource Business Center. See ["Registering with CyberSource," page 14](#). Samsung requires the CSR file in order to encrypt sensitive payment data; it contains an identifier and public key.

- Step 3** Using the [Samsung Pay Partner Portal](#), upload the CSR file.
- Step 4** Enter an application name and a package name.
- Step 5** When you associate the CSR file with the application, Samsung generates a product ID.
- Step 6** Create login details for application developers on the [Samsung Pay Partner Portal](#).
- Step 7** Download and integrate the Samsung Pay SDK into your application. See ["Using the Samsung Pay SDK," page 18](#).

The SDK contains:

- A Javadoc
- The Samsung Pay SDK files *samsungpay.jar* and *sdk-v1.0.0.jar*

- A sample app
- The branding guide
- Image files

Step 8 Register a Samsung account ID and request a *debug-api-key* file using the [Samsung Pay Partner Portal](#). The *debug-api-key* file is valid for three months. See "Using the API Key," page 17.

**Note**

The Samsung account ID, the *debug-api-key*, and the product ID are used to validate your application so that you can use the Samsung Pay SDK for testing purposes.

Step 9 Submit your application for approval using the [Samsung Pay Partner Portal](#). Upload the final version of the Android Application Package (APK) file using the [Samsung Pay Partner Portal](#) and include screenshots of your checkout page displaying the Samsung Pay logo.

Registering with CyberSource

To register with CyberSource:

- Step 1** Log in to the Business Center:
- Create a CSR file for live transactions: <https://ebc.cybersource.com>
 - Create a CSR file for test transactions: <https://ebctest.cybersource.com>
- Step 2** Under Account Management in the left navigation panel, click Digital Payment Solutions.
- Step 3** Click **Sign Up**. Follow the steps to verify your account information and accept the ProductName Merchant Services Agreement.

Step 4 Register with CyberSource:

- a** Enter your Samsung partner ID that you obtained in [Step 2](#).
- b** Click **Generate CSR** to generate a Certificate Signing Request (CSR) file that is associated with your Samsung partner ID.



Only one CSR is permitted for each unique Samsung partner ID. If you modify your Samsung partner ID you must generate a new CSR.

- c** Submit the CSR file to Samsung.
-

Integrating the Samsung SDK

Creating a Project

To create a new project using Android Studio:

- Step 1** Download [Android Studio](#).
- Step 2** Open Android Studio and click **Start a new Android Studio project**.
- Step 3** In the New Project settings, enter the following:
- The name of your application.
 - The company domain.
 - To change the package name, click **Edit**. By default, Android Studio sets the last element of the project's package name to the name of your application.
- Step 4** Click **Next**.
- Step 5** In the Target Android Devices settings, choose the required API levels.
- Step 6** Click **Next**.
- Step 7** Choose the required activity and click **Finish**.
-

Integrating the Samsung Pay SDK

To integrate the Samsung Pay SDK:

- Step 1** Add the *samsungpay.jar* and *sdk-v1.0.0.jar* files to the *libs* folder of your Android project.
- Step 2** Choose **Gradle Scripts > build.gradle** and enter the dependencies shown below.

```
dependencies {
    compile files('libs/samsungpay.jar')
    compile files('libs/sdk-v1.0.0.jar')
}
```

- Step 3** Import the package.

```
import com.samsung.android.sdk.samsungpay;
```

Using the API Key

The API key is used to verify that your app (in debug mode or release mode) can use the Samsung Pay SDK APIs with the Samsung Pay application. To get the API key, you must create a *debug-api-key* file (Step 8) and include it in the *manifest* file.

To use the API key:

- Step 1** Include the API key in the *manifest* file with a custom tag. This enables the merchant app android *manifest* file to provide the `DebugMode`, `spay_debug_api_key` values as meta-data.

Example 1 Debug Mode

```
<meta-data
    android:name="debug_mode"
    android:value="Y" />
<meta-data
    android:name="spay_debug_api_key"
    android:value="asdfggkndkeiel7283094858" />
```

Example 2 Release Mode

```
<meta-data
    android:name="debug_mode"
    android:value="N" />
```

Using the Samsung Pay SDK

Eligibility

Initialize the `SSamsungPay` class to verify that your application is eligible for Samsung Pay and to display the Samsung Pay button to the customer (refer to branding guidelines).

The `SSamsungPay` class provides the following API methods:

- `initialize()`—initializes the Samsung Pay SDK and verifies eligibility for Samsung Pay, including the device, software, and business area.



Request the `initialize()` API method of the `SSamsungPay` class before using the Samsung Pay SDK.

- `getVersionCode()`—retrieves the version number of the Samsung Pay SDK as an integer.
- `getVersionName()`—retrieves the version name of the Samsung Pay SDK as a string.

After the `initialize()` API method request is successful, display the Samsung Pay button to the customer.

If the `initialize()` API method request fails, the method displays a `SsdkUnsupportedException` or `NullPointerException` error.

- `SsdkUnsupportedException`—the device is not a Samsung device or does not support the Samsung Pay package.
- `NullPointerException`—the context passed is null.

Example 3 Samsung Pay Class

```

SSamsungPay spay = new SSamsungPay();
try {
    spay.initialize(mContext);
} catch (SsdkUnsupportedException e1) {
    e1.printStackTrace();
    pay_button.setVisibility(View.INVISIBLE);
}

```

Payment Request

Initiating a Payment

To initiate a payment:

Step 1 Include the following fields in the `PaymentInfo` class:

If the required fields are not included, you receive a `NullPointerException` error.

Important

- Merchant Name—the merchant name as it appears on the payment sheet of Samsung Pay and customer's bank statement. This field is required.
 - Amount—this field is required.
 - Payment Protocol—3D Secure. This field is required.
 - Permitted Card Brands—specify the card brands that are supported such as Visa, Mastercard, or American Express. This field is required.
 - Merchant ID
 - Order Number
 - Shipping Address—this field is required if `SEND_SHIPPING` or `NEED_BILLING_AND_SEND_SHIPPING` is set for `AddressVisibilityOption`.
 - Address Visibility Option
 - Card Holder Name
 - Recurring Option
-

Example 4 Transaction Request Structure

```

private PaymentInfo makeTransactionDetails() {
    // Supported card brands
    ArrayList<CardInfo.Brand> brandList = new ArrayList<CardInfo.Brand>();
    if (visaBrand.isChecked())
        brandList.add(CardInfo.Brand.VISA);
    if (mcBrand.isChecked())
        brandList.add(CardInfo.Brand.Mastercard);
    if (amexBrand.isChecked())
        brandList.add(CardInfo.Brand.AMERICANEXPRESS);

    // Basic payment information
    PaymentInfo paymentReq = new PaymentInfo.Builder()
        .setMerchantId("merchantID")
        .setMerchantName("Test").setAmount(getAmount())
        .setShippingAddress(getShippingAddressInfo())
        .setOrderNumber(orderNoView.getText().toString())
        .setPaymentProtocol(PaymentProtocol.PROTOCOL_3DS)
        .setAddressInPaymentSheet(AddressInPaymentSheet.DO_NOT_SHOW)
        .setAllowedCardBrands(brandList) .setRecurringEnabled(isRecurring)
        .setCardHolderNameEnabled(isCardHolderNameRequired)
        .build();
    return paymentReq;
}

// Add shipping address details
private Address getShippingAddressInfo() {
    Address address = new Address.Builder()
        .setAddressee(name.getText().toString())
        .setAddressLine1(addLine1.getText().toString())
        .setAddressLine2(addline2.getText().toString())
        .setCity(city.getText().toString())
        .setState(state.getText().toString())
        .setCountryCode(country.getSelectedItem().toString())
        .setPostalCode(zip.getText().toString()).build(); return address;
}

// Add amount details private Amount getAmount() {
    Amount amount = new Amount.Builder()
        .setCurrencyCode(currencyType.getSelectedItem().toString())
        .setItemTotalPrice(productPrice.getText().toString())
        .setShippingPrice(shippingPrice.getText().toString())
        .setTax(taxPrice.getText().toString())
        .setTotalPrice(totalAmount.getText().toString()).build();
    return amount;
}

```

Requesting a Payment

To request a payment:

Step 1 Use the `startSamsungPay()` API method in the `PaymentManager` class.

The `PaymentManager` class includes the following API methods:

- `startSamsungPay()`—requests to initiate payment with Samsung Pay.
- `updateAmount()`—updates the transaction amount if shipping address or card information is updated by Samsung Pay.
- `updateAmountFailed()`—returns an error code when the new amount cannot be updated because of a wrong address.

Step 2 Request the `startSamsungPay()` API method and include the following data:

- `PaymentInfo`—the `paymentInfo` structure, which contains payment information.
- `PID`—the product ID created in the Samsung Pay Partner Portal. See ["Registration," page 13](#).
- `StatusListener`—the result of the payment request is delivered to `StatusListener`. This listener should be registered before calling the `startSamsungPay()` API method.

When you request the `startSamsungPay()` API method, the Samsung Pay online payment sheet is displayed on the screen of your application. The customer selects a registered card for payment and can also update the billing and shipping address.

The payment reply is delivered as one of the following events to `StatusListener`:

- `onSuccess()`—this event is requested when Samsung Pay confirms the payment. It includes `encryptedPaymentCredential` in JSON format. See [Table 2, "Encrypted Payment Credential," on page 22](#).
 - `onFailure()`—this event is requested when the transaction fails. It returns an error code and error message.
-

Example 5 Request startSamsungPay() API Method

```

public void onPayButtonClicked(View v) {
    // Call startSamsungPay() method of PaymentManager class.
    // To create a transaction request for makeTransactionDetails() in
    // the following code, see Example 4, "Transaction Request Structure,"
    // on page 20.
    try {
        mPaymentManager.startSamsungPay(makeTransactionDetails(), "enter
        product ID",
mStatusListener);
    } catch (NullPointerException e) {
        e.printStackTrace();
    }
}

private PaymentManager.StatusListener mStatusListener = new
PaymentManager.StatusListener() {
    @Override
    public void onFailure(int errCode, String msg) {
        Log.d(TAG, " onFailed ");
    }
    @Override
    public void onSuccess(PaymentInfo arg0, String result) {
        Log.d(TAG, "onSuccess ");
    }
};

```

Table 2 Encrypted Payment Credential

| Payment Credential | Description |
|---------------------------------|--|
| method | Payment protocol: 3D Secure. |
| merchant_ref | Merchant reference code. |
| billing_address.street | Number, street name. |
| billing_address.state_province | Two letter state code. |
| billing_address.zip_postal_code | Five character zip code. |
| billing_address.city | City name. |
| billing_address.county | Two letter country code. |
| 3ds.type | S for Samsung Pay. Encrypted. |
| 3ds.version | Current version 100. Encrypted. |
| 3ds.data | Base64 encoded payment data. Encrypted. |

For information on how to decrypt the encrypted payment credential, see:

<https://pay.samsung.com/developers>

Authorizing a Payment

**Note**

Your payment processor can include additional API reply fields that are not documented in this guide. See *Credit Card Services Using the SCMP API* ([PDF](#) | [HTML](#)) for detailed descriptions of additional API reply fields.

Merchant Decryption

Visa Transaction

To request an authorization for a Visa transaction:

**Note**

See "API Request Fields," page 40 and "API Reply Fields," page 47 for detailed field descriptions.

- Step 1** Set the **customer_cc_number** field to the payment network token value.
- Step 2** Set the **customer_cc_expmo** and **customer_cc_expyr** values to the payment network token expiration date values.
- Step 3** Set the **cavv** field to the 3D Secure cryptogram of the payment network token.
- Step 4** Set the **network_token_cryptogram** field to the network token cryptogram.
- Step 5** Set the **payment_network_token_transaction_type** field to 1.
- Step 6** Set the **e_commerce_indicator** field to `internet`.
- Step 7** Set the **payment_solution** field to 008.

Example 6 Merchant Decryption Authorization Request (Visa)

```

bill_address1=111 S. Division St.
bill_address2=Suite 123
bill_city=Ann Arbor
bill_country=US
bill_state=MI
bill_zip=48104-2201
currency=usd
customer_cc_expmo=12
customer_cc_expyr=2021
customer_cc_number=xxxx100000000xxxx
customer_email=demo@example.com
customer_firstname=James
customer_ipaddress=66.123.123.2
customer_lastname=Smith
customer_phone=999-999-9999
e_commerce_indicator=internet
grand_total_amount=100.00
ics_applications=ics_auth
merchant_id=demomerchant
merchant_ref_number=demorefnum
cavv=ABCDEFabcdefABCDEFabcdef0987654321234567
xid=1234567890987654321ABCDEFabcdefABCDEF123
payment_network_token_transaction_type=1
solution_type=008

```

Example 7 Merchant Decryption Authorization Reply (Visa)

```

request_token=Ahj/7wSR5C/kX6302hAKIkGLNkwcsmrSHH1U5tGHRT/hHgzc8BT/hHgk
currency=usd
request_id=4465837560045000001541
auth_rflag=SOK
ics_rmsg=Request was processed successfully.
auth_auth_amount=100.00
auth_rcode=1
auth_trans_ref_no=13209254CGJSMQCQ
auth_auth_code=888888
auth_rmsg=Request was processed successfully.
ics_rflag=SOK
auth_auth_response=100
auth_avs_raw=I1
auth_auth_time=2015-11-03T204917Z
merchant_ref_number=demorefnum
ics_rcode=1

```

Mastercard Transaction

To request an authorization for a Mastercard transaction:

**Note**

See "API Request Fields," page 40 and "API Reply Fields," page 47 for detailed field descriptions.

- Step 1** Set the **customer_cc_number** field to the payment network token value.
 - Step 2** Set the **customer_cc_expmo** and **customer_cc_expyr** values to the payment network token expiration date values.
 - Step 3** Set the **ucaf_authentication_data** field to the 3D Secure cryptogram of the payment network token.
 - Step 4** Set the **network_token_cryptogram** field to the network token cryptogram.
 - Step 5** Set the **ucaf_collection_indicator** field to 2.
 - Step 6** Set the **payment_network_token_transaction_type** field to 1.
 - Step 7** Set the **e_commerce_indicator** field to `spa`.
 - Step 8** Set the **payment_solution** field to 008.
-

Example 8 Merchant Decryption Authorization Request (Mastercard)

```

bill_address1=111 S. Division St.
bill_address2=Suite 123
bill_city=Ann Arbor
bill_country=US
bill_state=MI
bill_zip=48104-2201
currency=usd
customer_cc_expmo=12
customer_cc_expyr=2021
customer_cc_number=xxxx55555555xxxx
customer_email=demo@example.com
customer_firstname=James
customer_ipaddress=66.123.123.2
customer_lastname=Smith
customer_phone=999-999-9999
grand_total_amount=100.00
ics_applications=ics_auth
merchant_id=demomerchant
merchant_ref_number=demorefnum
ucaf_authentication_data=ABCDEFabcdefABCDEFabcdef0987654321234567
ucaf_collection_indicator=2
payment_network_token_transaction_type=1
solution_type=008

```

Example 9 Merchant Decryption Authorization Reply (Mastercard)

```

request_token=Ahj/7wSR5C/p6oJEyIlgKIkGLNkwcsmrWHH1U5tGHST/hHgzdACT/hVB3c
currency=usd
request_id=4465838340055000001541
auth_rflag=SOK
ics_rmsg=Request was processed successfully.
auth_auth_amount=100.00
auth_rcode=1
auth_trans_ref_no=13209255CGJSMQCR
auth_auth_code=888888
auth_rmsg=Request was processed successfully.
ics_rflag=SOK
auth_auth_response=100
auth_avs_raw=I1
auth_auth_time=2015-11-03T205035Z
merchant_ref_number=demorefnum
ics_rcode=1

```

American Express Transaction

To request an authorization for an American Express transaction:

**Note**

See "API Request Fields," page 40 and "API Reply Fields," page 47 for detailed field descriptions.

- Step 1** Set the **customer_cc_number** field to the payment network token value.
 - Step 2** Set the **customer_cc_expmo** and **customer_cc_expyr** values to the payment network token expiration date values.
 - Step 3** Set the **cavv** field to the 3D Secure cryptogram of the payment network token.
-

**Important**

Include the whole 20-byte cryptogram in the **cavv** field. For a 40-byte cryptogram, split the cryptogram into two 20-byte binary values (block A and block B). Set the **cavv** field to the block A value and set the **xid** field to the block B value.

- Step 4** Set the **network_token_cryptogram** field to the network token cryptogram.
 - Step 5** Set the **payment_network_token_transaction_type** field to 1.
 - Step 6** Set the **e_commerce_indicator** field to `aesk`.
 - Step 7** Set the **payment_solution** field to `008`.
-

Example 10 Merchant Decryption Authorization Request (American Express)

```

bill_address1=111 S. Division St.
bill_address2=Suite 123
bill_city=Ann Arbor
bill_country=US
bill_state=MI
bill_zip=48104-2201
currency=usd
customer_cc_expmo=12
customer_cc_expyr=2021
customer_cc_number=xxxx82246310xxxx
customer_email=demo@example.com
customer_firstname=James
customer_ipaddress=66.123.123.2
customer_lastname=Smith
customer_phone=999-999-9999
grand_total_amount=100.00
ics_applications=ics_auth
merchant_id=demomerchant
merchant_ref_number=demorefnum
cavv=ABCDEFabcdefABCDEFabcdef0987654321234567
xid=1234567890987654321ABCDEFabcdefABCDEF123
payment_network_token_transaction_type=1
solution_type=008

```

Example 11 Merchant Decryption Authorization Reply (American Express)

```

request_token=Ahj/7wSR5C/wGXKw1xAKIkGLNkwcmraHH1U5tGHaT/hHgzecDT/h6BBL
currency=usd
request_id=4465839210285000001541
auth_rflag=SOK
ics_rmsg=Request was processed successfully.
auth_auth_amount=100.00
auth_rcode=1
auth_trans_ref_no=13209256CGJSMQCZ
auth_auth_code=888888
auth_rmsg=Request was processed successfully.
ics_rflag=SOK
auth_auth_response=100
auth_avs_raw=I1
auth_auth_time=2015-11-03T205202Z
merchant_ref_number=demorefnum
ics_rcode=1

```

JCB Transaction

To request an authorization for a JCB transaction:

**Note**

See "API Request Fields," page 40 and "API Reply Fields," page 47 for detailed field descriptions.

- Step 1** Set the **customer_cc_number** field to the payment network token value.
 - Step 2** Set the **customer_cc_expmo** and **customer_cc_expyr** fields to the payment network token expiration date values.
 - Step 3** Set the **cavv** field to the 3D Secure cryptogram of the payment network token.
 - Step 4** Set the **network_token_cryptogram** field to the network token cryptogram.
 - Step 5** Set the **payment_network_token_transaction_type** field to 1.
 - Step 6** Set the **eci_raw** field to the ECI value contained in the Samsung Pay response payload.
 - Step 7** Set the **payment_solution** field to 008.
-

Example 12 Merchant Decryption Authorization Request (JCB)

```

bill_address1=123 Main Street
bill_address2=Suite 12345
bill_city=Small Town
bill_country=US
bill_state=CA
bill_zip=98765
card_type=007
currency=usd
customer_cc_expmo=12
customer_cc_expyr=2031
customer_cc_number=xxxx11111111xxxx
customer_email=js@example.com
customer_firstname=Jane
customer_lastname=Smith
customer_phone=999-999-9999
eci_raw=05
grand_total_amount=100.00
ics_applications=ics_auth
merchant_id=med123
cavv=EHuWW9PiBkWvqE5juRwDzAUFBAk=
payment_network_token_transaction_type=1
payment_solution=008

```

Example 13 Merchant Decryption Authorization Reply (JCB)

```

auth_auth_amount=100.00
auth_auth_avs=X
auth_auth_code=888888
auth_auth_response=100
auth_avs_raw=I1
auth_rcode=1
auth_rflag=SOK
auth_rmsg=Request was processed successfully.
auth_trans_ref_no=15356268CR2XF23X
currency=USD
ics_rcode=1
ics_rflag=SOK
ics_rmsg=Request was processed successfully.
merchant_ref_number=ref123
request_id=4697369268106124601541
request_token=Ahj/7wSR/UoVmlbMmziHSZjMECT/h+KjMHSB04gwGA2dDjQoxQAAA6xdr

```

CyberSource Decryption

Visa Transaction

To request an authorization for a Visa transaction:



Note

See "API Request Fields," page 40 and "API Reply Fields," page 47 for detailed field descriptions.

Step 1 Set the **encrypted_payment_data** field to the value that was returned from Samsung Pay in the *3ds.data* block.

- a** Retrieve the payment data from Samsung Pay in JSON Web Encryption (JWE) format.
- b** Encode it in Base64.
- c** Retrieve the corresponding Key ID (KID) with encryption and set the values as:

```
{
  "publicKeyHash": "kid",
  "version": "100",
  "data": "encoded data from step b above"
}
```

- d** Encode the structure in Base64.
- e** Add the value to the **encrypted_payment_data** field.

Step 2 Set the **encrypted_payment_descriptor** field to `Rk1EPUNPTU1PTi5TQU1TVU5HLk1OQVBQL1BBWU1FTlQ=.`

Step 3 Set the **payment_network_token_transaction_type** field to 1.

Step 4 Set the **e_commerce_indicator** field to `internet`.

Step 5 Set the **payment_solution** field to 008.

Example 14 CyberSource Decryption Authorization Request (Visa)

```

bill_address1=111 S. Division St.
bill_address2=Suite 123
bill_city=Ann Arbor
bill_country=US
bill_state=MI
bill_zip=48104-2201
currency=usd
customer_email=demo@example.com
customer_firstname=James
customer_ipaddress=66.123.123.2
customer_lastname=Smith
customer_phone=999-999-9999
e_commerce_indicator=internet
grand_total_amount=100.00
ics_applications=ics_auth
merchant_id=demomerchant
merchant_ref_number=demorefnum
encrypted_payment_data=ABCDEFabcdefABCDEFabcdef0987654321234567
encrypted_payment_
descriptor=RklEPUNPTU1PTi5TQU1TVU5HLk1OQVBQL1BBWU1FTlQ=
payment_network_transaction_type=1
solution_type=008

```

Example 15 CyberSource Decryption Authorization Reply (Visa)

```

request_token=Ahj/7wSR5C/kX6302hAKIkGLNkwcsmrSHH1U5tGHRT/hHgzc8BT/hHgk
currency=usd
request_id=4465837560045000001541
auth_rflag=SOK
ics_rmsg=Request was processed successfully.
auth_auth_amount=100.00
auth_rcode=1
auth_trans_ref_no=13209254CGJSMQCQ
auth_auth_code=888888
auth_rmsg=Request was processed successfully.
ics_rflag=SOK
auth_auth_response=100
auth_avs_raw=I1
merchant_ref_number=demorefnum
ics_rcode=1
token_prefix=294672
token_suffix=4397
token_expirationMonth=08
token_expirationYear=2021

```

Mastercard Transaction

To request an authorization for a Mastercard transaction:



Note

See "API Request Fields," page 40 and "API Reply Fields," page 47 for detailed field descriptions.

Step 1 Set the **encrypted_payment_data** field to the value that was returned from Samsung Pay in the *3ds.data* block.

- a** Retrieve the payment data from Samsung Pay in JSON Web Encryption (JWE) format.
- b** Encode it in Base64.
- c** Retrieve the corresponding Key ID (KID) with encryption and set the values as:

```
{
  "publicKeyHash": "kid",
  "version": "100",
  "data": "encoded data from step b above"
}
```

- d** Encode the structure in Base64.
- e** Add the value to the **encrypted_payment_data** field.

Step 2 Set the **encrypted_payment_descriptor** field to
Rk1EPUNPTU1PTi5TQU1TVU5HLk1OQVBQ11BBWU1FT1Q=.

Step 3 Set the **payment_network_token_transaction_type** field to 1.

Step 4 Set the **e_commerce_indicator** field to *spa*.

Step 5 Set the **payment_solution** field to 008.

Example 16 CyberSource Decryption Authorization Request (Mastercard)

```

bill_address1=111 S. Division St.
bill_address2=Suite 123
bill_city=Ann Arbor
bill_country=US
bill_state=MI
bill_zip=48104-2201
currency=usd
customer_email=demo@example.com
customer_firstname=James
customer_ipaddress=66.123.123.2
customer_lastname=Smith
customer_phone=999-999-9999
e_commerce_indicator=spa
grand_total_amount=100.00
ics_applications=ics_auth
merchant_id=demomerchant
merchant_ref_number=demorefnum
encrypted_payment_data=ABCDEFabcdefABCDEFabcdef0987654321234567
encrypted_payment_
descriptor=RklEPUNPTU1PTi5TQU1TVU5HLk1OQVBQL1BBWU1FTlQ=
payment_network_transaction_type=1
solution_type=008

```

Example 17 CyberSource Decryption Authorization Reply (Mastercard)

```

request_token=Ahj/7wSR5C/p6oJEy1gKIkGLNkwcsmrWHH1U5tGHST/hHgzdACT/hVB3c
currency=usd
request_id=4465838340055000001541
auth_rflag=SOK
ics_rmsg=Request was processed successfully.
auth_auth_amount=100.00
auth_rcode=1
auth_trans_ref_no=13209255CGJSMQCR
auth_auth_code=888888
auth_rmsg=Request was processed successfully.
ics_rflag=SOK
auth_auth_response=100
auth_avs_raw=I1
auth_auth_time=2015-11-03T205035Z
merchant_ref_number=demorefnum
ics_rcode=1
token_prefix=128945
token_suffix=2398
token_expirationMonth=08
token_expirationYear=2021

```

American Express Transaction

To request an authorization for an American Express transaction:



Note

See "API Request Fields," page 40 and "API Reply Fields," page 47 for detailed field descriptions.

Step 1 Set the **encrypted_payment_data** field to the value that was returned from Samsung Pay in the *3ds.data* block.

- a** Retrieve the payment data from Samsung Pay in JSON Web Encryption (JWE) format.
- b** Encode it in Base64.
- c** Retrieve the corresponding Key ID (KID) with encryption and set the values as:

```
{
  "publicKeyHash": "kid",
  "version": "100",
  "data": "encoded data from step b above"
}
```

- d** Encode the structure in Base64.
- e** Add the value to the **encrypted_payment_data** field.

Step 2 Set the **encrypted_payment_descriptor** field to
Rk1EPUNPTU1PTi5TQU1TVU5HLk1OQVBQL1BBWU1FTlQ=.

Step 3 Set the **payment_network_token_transaction_type** field to 1.

Step 4 Set the **e_commerce_indicator** field to *aesk*.

Step 5 Set the **payment_solution** field to 008.

Example 18 CyberSource Decryption Authorization Request (American Express)

```

bill_address1=111 S. Division St.
bill_address2=Suite 123
bill_city=Ann Arbor
bill_country=US
bill_state=MI
bill_zip=48104-2201
card_type=003
currency=usd
customer_email=demo@example.com
customer_firstname=James
customer_ipaddress=66.123.123.2
customer_lastname=Smith
customer_phone=999-999-9999
e_commerce_indicator=aesk
grand_total_amount=100.00
ics_applications=ics_auth
merchant_id=demomerchant
merchant_ref_number=demorefnum
encrypted_payment_data=ABCDEFabcdefABCDEFabcdef0987654321234567
encrypted_payment_
descriptor=RklEPUNPTU1PTi5TQU1TVU5HLk1OQVBQL1BBWU1FTlQ=
payment_network_transaction_type=1
solution_type=008

```

Example 19 CyberSource Decryption Authorization Reply (American Express)

```

request_token=Ahj/7wSR5C/wGXKw1xAKIkGLNkwcsmraHH1U5tGHaT/hHgzecDT/h6BBL
currency=usd
request_id=4465839210285000001541
auth_rflag=SOK
ics_rmsg=Request was processed successfully.
auth_auth_amount=100.00
auth_rcode=1
auth_trans_ref_no=13209256CGJSMQCZ
auth_auth_code=888888
auth_rmsg=Request was processed successfully.
ics_rflag=SOK
auth_auth_response=100
auth_avs_raw=I1
auth_auth_time=2015-11-03T205202Z
merchant_ref_number=demorefnum
ics_rcode=1
token_prefix=593056
token_suffix=0842
token_expirationMonth=08
token_expirationYear=2021

```

JCB Transaction

To request an authorization for a JCB transaction:



Note

See "API Request Fields," page 40 and "API Reply Fields," page 47 for detailed field descriptions.

- Step 1** Set the **encrypted_payment_data** field to the Base64 encoded value obtained from the **paymentData** property of the **PKPaymentToken** object.
 - Step 2** Set the **encrypted_payment_descriptor** field to `RklEPUNPTU1PTi5TQU1TVU5Hlk1OQVBQL1BBWU1FTlQ=`.
 - Step 3** Set the **payment_solution** field to `008`.
-

Example 20 CyberSource Decryption Authorization Request (JCB)

```
bill_address1=123 Main Street
bill_address2=Suite 12345
bill_city=Small Town
bill_country=US
bill_state=CA
bill_zip=98765
card_type=007
currency=usd
customer_cc_expmo=12
customer_cc_expyr=2031
customer_cc_number=xxxx55555555xxxx
customer_email=js@example.com
customer_firstname=Jane
customer_lastname=Smith
customer_phone=999-999-9999
eci_raw=05
grand_total_amount=100.00
ics_applications=ics_auth
merchant_id=med123
cavv=EHuWW9PiBkWvqE5juRwDzAUFBAk=
payment_network_token_transaction_type=1
payment_solution=008
```

Example 21 CyberSource Decryption Authorization Reply (JCB)

```

auth_auth_amount=100.00
auth_auth_avs=X
auth_auth_code=888888
auth_auth_response=100
auth_avs_raw=I1
auth_rcode=1
auth_rflag=SOK
auth_rmsg=Request was processed successfully.
auth_trans_ref_no=15356268CR2XF23X
currency=USD
ics_rcode=1
ics_rflag=SOK
ics_rmsg=Request was processed successfully.
merchant_ref_number=ref123
request_id=4697369268106124601541
request_token=Ahj/7wSR/UoVmlbMmziHSZjMECT/h+KjMHSB04gwGA2dDjQoxQAAA6xdr

```

Additional CyberSource Services

Refer to *Credit Card Services Using the SCMP API* ([PDF](#) | [HTML](#)) for information on how to request these follow-on services.

Table 3 CyberSource Services

| CyberSource Service | Description |
|------------------------|--|
| Capture | A follow-on service that uses the request ID returned from the previous authorization. The request ID links the capture to the authorization. This service transfers funds from the customer's account to your bank and usually takes two to four days to complete. |
| Sale | A sale is a bundled authorization and capture. Request the authorization and capture services at the same time. CyberSource processes the capture immediately. |
| Authorization Reversal | A follow-on service that uses the request ID returned from the previous authorization. An authorization reversal releases the hold that the authorization placed on the customer's credit card funds. Use this service to reverse an unnecessary or undesired authorization. |

API Fields

Data Type Definitions

| Data Type | Description |
|---------------------|--|
| Date and time | Format is YYYY-MM-DDThhmmssZ, where: <ul style="list-style-type: none"> ■ T separates the date and the time. ■ Z indicates Coordinated Universal Time (UTC), which equals Greenwich Mean Time (GMT). Example: 2016-08-11T22:47:57Z equals August 11, 2016, at 22:47:57 (10:47:57 p.m.) |
| Decimal | Number that includes a decimal point Examples: 23.45, -0.1, 4.0, 90809.0468 |
| Integer | Whole number {..., -3, -2, -1, 0, 1, 2, 3, ...} |
| Nonnegative integer | Whole number greater than or equal to zero {0, 1, 2, 3, ...} |
| Positive integer | Whole number greater than zero {1, 2, 3, ...} |
| String | Sequence of letters, numbers, spaces, and special characters |

Relaxed Requirements for Address Data

To enable relaxed requirements for address data and expiration date, contact CyberSource Customer Support to have your account configured for this feature. For details about relaxed requirements, see the [Relaxed Requirements for Address Data and Expiration Date](#) page.

API Request Fields



Note

Unless otherwise noted, all field names are case sensitive and all fields accept special characters such as @, #, and %.

Table 4 API Request Fields

| Field | Description | Used By: Required (R) or Optional (O) | Data Type (Length) |
|--|---|---|-----------------------|
| bill_address1 | <p>First line of the billing street address.</p> <p>Important It is your responsibility to determine whether a field is required for the transaction you are requesting.</p> <p>See "Relaxed Requirements for Address Data," page 39.</p> | ics_auth (See description) | String (60) |
| bill_address2 | <p>Additional address information.</p> <p>Example Attention: Accounts Payable</p> | ics_auth (R) | String (60) |
| bill_city | <p>City of the billing address.</p> <p>Important It is your responsibility to determine whether a field is required for the transaction you are requesting.</p> <p>See "Relaxed Requirements for Address Data," page 39.</p> | ics_auth (See description) | String (50) |
| bill_country | <p>Country of the billing address. Use the two-character <i>ISO Standard Country Codes</i>.</p> <p>Important It is your responsibility to determine whether a field is required for the transaction you are requesting.</p> <p>See "Relaxed Requirements for Address Data," page 39.</p> | ics_auth (See description) | String (2) |
| <p>1 The TC 33 Capture file contains information about the purchases and refunds that a merchant submits to CyberSource. CyberSource through VisaNet creates the TC 33 Capture file at the end of the day and sends it to the merchant's acquirer, who uses this information to facilitate end-of-day clearing processing with payment card companies.</p> | | | |

Table 4 API Request Fields (Continued)

| Field | Description | Used By: Required (R) or Optional (O) | Data Type (Length) |
|--|--|---|-----------------------|
| bill_state | <p>State or province of the billing address. For an address in the U.S. or Canada, use the State, Province, and Territory Codes for the United States and Canada.</p> <p>Important It is your responsibility to determine whether a field is required for the transaction you are requesting.</p> <p>See "Relaxed Requirements for Address Data," page 39.</p> | ics_auth (See description) | String (2) |
| bill_zip | <p>Postal code for the billing address. The postal code must consist of 5 to 9 digits.</p> <p>When the billing country is the U.S., the 9-digit postal code must follow this format: [5 digits][dash][4 digits]</p> <p>Example 12345-6789</p> <p>When the billing country is Canada, the 6-digit postal code must follow this format: [alpha][numeric][alpha][space][numeric][alpha][numeric]</p> <p>Example A1B 2C3</p> <p>Important It is your responsibility to determine whether a field is required for the transaction you are requesting.</p> <p>See "Relaxed Requirements for Address Data," page 39.</p> | ics_auth (See description) | String (9) |
| <p>1 The TC 33 Capture file contains information about the purchases and refunds that a merchant submits to CyberSource. CyberSource through VisaNet creates the TC 33 Capture file at the end of the day and sends it to the merchant's acquirer, who uses this information to facilitate end-of-day clearing processing with payment card companies.</p> | | | |

Table 4 API Request Fields (Continued)

| Field | Description | Used By: Required (R) or Optional (O) | Data Type (Length) |
|--|--|---|--------------------------|
| cavv | <p>Visa Cryptogram for payment network tokenization transactions. The value for this field must be 28-character base64 or 40-character hex binary. All cryptograms use one of these formats.</p> <p>American Express For a 20-byte cryptogram, set this field to the cryptogram for payment network tokenization transactions. For a 40-byte cryptogram, set this field to block A of the cryptogram for payment network tokenization transactions. The value for this field must be 28-character base64 or 40-character hex binary. All cryptograms use one of these formats.</p> | ics_auth (R) | String (40) |
| currency | Currency used for the order: USD | ics_auth (R) | String (5) |
| customer_cc_expmo | Two-digit month in which the payment network token expires. Format: MM. Possible values: 01 through 12. | ics_auth (R) | String (2) |
| customer_cc_expyr | Four-digit year in which the payment network token expires. Format: YYYY. | ics_auth (R) | Nonnegative integer (4) |
| customer_cc_number | The payment network token value. This value is obtained by decrypting the customer's encrypted payment data. | ics_auth (R) | Nonnegative integer (20) |
| customer_email | Customer's email address. Important It is your responsibility to determine whether a field is required for the transaction you are requesting. See "Relaxed Requirements for Address Data," page 39. | ics_auth (See description) | String (255) |
| <p>1 The TC 33 Capture file contains information about the purchases and refunds that a merchant submits to CyberSource. CyberSource through VisaNet creates the TC 33 Capture file at the end of the day and sends it to the merchant's acquirer, who uses this information to facilitate end-of-day clearing processing with payment card companies.</p> | | | |

Table 4 API Request Fields (Continued)

| Field | Description | Used By: Required (R) or Optional (O) | Data Type (Length) |
|--|--|---|-----------------------|
| customer_firstname | Customer's first name. For a credit card transaction, this name must match the name on the card. Important It is your responsibility to determine whether a field is required for the transaction you are requesting. See "Relaxed Requirements for Address Data," page 39. | ics_auth (See description) | String (60) |
| customer_ipaddress | Customer's IP address. | ics_auth (O) | String (15) |
| customer_lastname | Customer's last name. For a credit card transaction, this name must match the name on the card. Important It is your responsibility to determine whether a field is required for the transaction you are requesting. See "Relaxed Requirements for Address Data," page 39. | ics_auth (See description) | String (60) |
| customer_phone | Customer's phone number. CyberSource recommends that you include the country code when the order is from outside the U.S. | ics_auth (O) | String (15) |
| eci_raw | Raw electronic commerce indicator (ECI). | ics_auth | String (2) |
| directory_server_transaction_id | Identifier generated during the authentication transaction by the Mastercard Directory Server and passed back with the authentication results. | ics_auth (O) | String (36) |
| e_commerce_indicator | For a payment network tokenization transaction. Possible values: <ul style="list-style-type: none"> ■ <code>aesk</code>: American Express card type ■ <code>spa</code>: Mastercard card type ■ <code>internet</code>: Visa card type | ics_auth (O) | String (20) |
| <p>1 The TC 33 Capture file contains information about the purchases and refunds that a merchant submits to CyberSource. CyberSource through VisaNet creates the TC 33 Capture file at the end of the day and sends it to the merchant's acquirer, who uses this information to facilitate end-of-day clearing processing with payment card companies.</p> | | | |

Table 4 API Request Fields (Continued)

| Field | Description | Used By: Required (R) or Optional (O) | Data Type (Length) |
|------------------------------|--|---|-----------------------|
| encrypted_payment_data | The encrypted payment data value. If you are using the CyberSource Decryption option, populate this field with the encrypted payment data value returned from Samsung Pay in the <i>3ds.data</i> block. See " CyberSource Decryption ," page 31. | ics_auth (R) | |
| encrypted_payment_descriptor | Format of the encrypted payment data. The value for Samsung Pay is Rk1EPUNPTU1PTi5TQU1TVU5HLk1O QVBQL1BBWU1FT1Q= | ics_auth (R) | |
| grand_total_amount | Grand total for the order. This value cannot be negative. You can include a decimal point (.), but you cannot include any other special characters. CyberSource truncates the amount to the correct number of decimal places. | ics_auth (R) | Decimal (15) |
| ics_applications | CyberSource services to process for the request: ics_auth | ics_auth (R) | String (255) |
| merchant_id | Your CyberSource merchant ID. Use the same merchant ID for evaluation, testing, and production. | ics_auth (R) | String (30) |
| merchant_ref_number | Merchant-generated order reference or tracking number. CyberSource recommends that you send a unique value for each transaction so that you can perform meaningful searches for the transaction. For information about tracking orders, see <i>Getting Started with CyberSource Advanced for the SCMP API</i> (PDF HTML). | ics_auth (R) | String (50) |
| network_token_cryptogram | Token authentication verification value cryptogram. For token-based transactions with 3D Secure or SecureCode, you must submit both types of cryptograms: network token and 3D Secure/SecureCode. The value for this field must be 28-character Base64 or 40-character hex binary. All cryptograms use one of these formats. | ics_auth (O) | String (40) |

1 The TC 33 Capture file contains information about the purchases and refunds that a merchant submits to CyberSource. CyberSource through VisaNet creates the TC 33 Capture file at the end of the day and sends it to the merchant's acquirer, who uses this information to facilitate end-of-day clearing processing with payment card companies.

Table 4 API Request Fields (Continued)

| Field | Description | Used By: Required (R) or Optional (O) | Data Type (Length) |
|--|--|---|-----------------------|
| pa_specification_version | The 3D Secure version that you used for Secured Consumer Authentication (SCA); for example, 3D Secure 1.0.2 or 2.0.0. | ics_auth (O) | String (20) |
| payment_network_token_assurance_level | Confidence level of the tokenization. This value is assigned by the token service provider. Note This field is supported only for FDC Nashville Global. | ics_auth (O) | String (2) |
| payment_network_token_device_tech_type | Type of technology used in the device to store token data. Possible value: 002: Host card emulation (HCE) Emulation of a smart card by using software to create a virtual and exact representation of the card. Sensitive data is stored in a database that is hosted in the cloud. For storing payment credentials, a database must meet very stringent security requirements that exceed PCI DSS. Note This field is supported only for FDC Compass. | ics_auth (O) | Integer (3) |
| payment_network_token_requestor_id | Value that identifies your business and indicates that the cardholder's account number is tokenized. This value is assigned by the token service provider and is unique within the token service provider's database. Note This field is supported only for FDC Nashville Global and Chase Paymentech Solutions. | ics_auth (O) | Integer (1) |
| payment_network_token_transaction_type | Type of transaction that provided the token data. This value does not specify the token service provider; it specifies the entity that provided you with information about the token. Set the value for this field to 1. | ics_auth (R) | String (1) |
| <p>1 The TC 33 Capture file contains information about the purchases and refunds that a merchant submits to CyberSource. CyberSource through VisaNet creates the TC 33 Capture file at the end of the day and sends it to the merchant's acquirer, who uses this information to facilitate end-of-day clearing processing with payment card companies.</p> | | | |

Table 4 API Request Fields (Continued)

| Field | Description | Used By: Required (R) or Optional (O) | Data Type (Length) |
|---------------------------|---|---|------------------------------|
| payment_solution | Identifies Samsung Pay as the payment solution that is being used for the transaction: Set the value for this field to 008. Note This unique ID differentiates digital solution transactions within the CyberSource platform for reporting purposes. | ics_auth (R) | String (3) |
| ucaf_authentication_data | Cryptogram for payment network tokenization transactions with Mastercard. | ics_auth (R) | String (32) |
| ucaf_collection_indicator | Required field for payment network tokenization transactions with Mastercard. Set the value for this field to 2. | ics_auth (R) | String with numbers only (1) |
| xid | Visa Cryptogram for payment network tokenization transactions. The value for this field must be 28-character base64 or 40-character hex binary. All cryptograms use one of these formats. American Express For a 20-byte cryptogram, set this field to the cryptogram for payment network tokenization transactions. For a 40-byte cryptogram, set this field to block A of the cryptogram for payment network tokenization transactions. See " Merchant Decryption ," page 23. The value for this field must be 28-character base64 or 40-character hex binary. All cryptograms use one of these formats. | ics_auth (R) | String (40) |

1 The TC 33 Capture file contains information about the purchases and refunds that a merchant submits to CyberSource. CyberSource through VisaNet creates the TC 33 Capture file at the end of the day and sends it to the merchant's acquirer, who uses this information to facilitate end-of-day clearing processing with payment card companies.

Offer-Level Fields

Table 5 Offer-Level Fields

| Field | Description | Used By: Required (R) or Optional (O) | Data Type (Length) |
|----------------------|---|---|-----------------------|
| amount | Per-item price of the product. This value cannot be negative. You can include a decimal point (.), but you cannot include any other special characters. | ics_auth (See description) | Decimal (15) |
| merchant_product_sku | Identification code for the product. This field is required when the product_code value is not <code>default</code> or one of the values related to shipping and/or handling. | ics_auth (See description) | String (255) |
| product_code | Type of product. This value is used to determine the product category: electronic, handling, physical, service, or shipping. The default is <code>default</code> . | ics_auth (See description) | String (255) |
| product_name | Name of the product. This field is required when the product_code value is not <code>default</code> or one of the values related to shipping and/or handling. | ics_auth (See description) | String (255) |
| quantity | The default is 1. This field is required when the product_code value is not <code>default</code> or one of the values related to shipping and/or handling. | ics_auth (See description) | Integer (10) |
| tax_amount | Total tax to apply to the product. This value cannot be negative. | ics_auth (See description) | String (15) |

API Reply Fields



Important

Because CyberSource can add reply fields, reply codes, and reply flags at any time:

- You must parse the reply data according to the names of the fields instead of the field order in the reply. For more information about parsing reply fields, see the documentation for your client.
- Your error handler should be able to process new reply codes and reply flags without problems.
- Your error handler should use the **ics_rcode** field to determine the result if it receives a reply flag that it does not recognize.



Note

Your payment processor can include additional API reply fields that are not documented in this guide. See *Credit Card Services Using the SCMP API* ([PDF](#) | [HTML](#)) for detailed descriptions of additional API reply fields.

Table 6 Reply Fields

| Field | Description | Returned By | Data Type & Length |
|--------------------|---|-------------|--------------------|
| auth_auth_amount | Amount that was authorized. | ics_auth | Decimal (15) |
| auth_auth_avs | AVS result code. See <i>Credit Card Services Using the SCMP API</i> (PDF HTML) for a detailed list of AVS values. | ics_auth | String (1) |
| auth_auth_code | Authorization code. Returned only when the processor returns this value. | ics_auth | String (7) |
| auth_auth_response | For most processors, this value is the error message sent directly from the bank. Returned only when the processor returns this value. | ics_auth | String (10) |
| auth_auth_time | Time of authorization in UTC. See " Data Type Definitions ," page 39. | ics_auth | Date and time (20) |
| auth_avs_raw | AVS result code sent directly from the processor. Returned only when the processor returns this value. | ics_auth | String (10) |
| auth_rcode | Indicates whether the service request was successful. Possible values: <ul style="list-style-type: none"> ■ -1: An error occurred. ■ 0: The request was declined. ■ 1: The request was successful. | ics_auth | Integer (1) |
| auth_rflag | One-word description of the result of the entire request. See <i>Credit Card Services Using the SCMP API</i> (PDF HTML) for a detailed list of rflag values. | ics_auth | String (50) |
| auth_rmsg | Message that explains the reply flag auth_rflag . Do not display this message to the customer, and do not use this field to write an error handler. | ics_auth | String (255) |
| auth_trans_ref_no | Reference number for the transaction. This value is not returned for all processors. | ics_auth | String (60) |

Table 6 Reply Fields (Continued)

| Field | Description | Returned By | Data Type & Length |
|---------------------------------|--|----------------------------------|--------------------|
| card_suffix | Last four digits of the cardholder's account number. This field is returned only for tokenized transactions. You can use this value on the receipt that you give to the cardholder. This field is returned only for FDC Nashville Global. | ics_auth | String (4) |
| currency | Currency used for the order. For the possible values, see the ISO Standard Currency Codes . | ics_auth | String (5) |
| directory_server_transaction_id | Identifier generated during the authentication transaction by the Mastercard Directory Server and passed back with the authentication results. | pa_enroll (O) pa_validate (O) | String (36) |
| ics_rcode | Indicates whether the service request was successful. Possible values: <ul style="list-style-type: none"> ■ -1: An error occurred. ■ 0: The request was declined. ■ 1: The request was successful. | ics_auth | Integer (1) |
| ics_rflag | One-word description of the result of the entire request. See Credit Card Services Using the SCMP API (PDF HTML) for a detailed list of rflag values. | ics_auth | String (50) |
| ics_rmsg | Message that explains the reply flag ics_rflag . Do not display this message to the customer, and do not use this field to write an error handler. | ics_auth | String (255) |
| merchant_ref_number | Order reference or tracking number that you provided in the request. If you included multi-byte characters in this field in the request, the returned value might include corrupted characters. | ics_auth | String (50) |
| request_id | Identifier for the request generated by the client. | ics_auth | String (26) |

Table 6 Reply Fields (Continued)

| Field | Description | Returned By | Data Type & Length |
|------------------------|--|--------------------|-------------------------------|
| request_token | Request token data created by CyberSource for each reply. The field is an encoded string that contains no confidential information such as an account or card verification number. The string can contain a maximum of 256 characters. | ics_auth | String (256) |
| token_expiration_month | Month in which the token expires. CyberSource includes this field in the reply message when it decrypts the payment blob for the tokenized transaction. Format: MM. Possible values: 01 through 12. | ics_auth | String (2) |
| token_expiration_year | Year in which the token expires. CyberSource includes this field in the reply message when it decrypts the payment blob for the tokenized transaction. Format: YYYY. | ics_auth | String (4) |
| token_prefix | First six digits of token. CyberSource includes this field in the reply message when it decrypts the payment blob for the tokenized transaction. | ics_auth | String (6) |
| token_suffix | Last four digits of token. CyberSource includes this field in the reply message when it decrypts the payment blob for the tokenized transaction. | ics_auth | String (4) |