Apple Pay

Simple Order API

Visa Platform Connect

cybersource
A Visa Solution

Developer Guide

Version: 25.04.01

Contents

Apple Pay Developer Guide **5**
Recent Revisions to This Document 7
Introduction to Apple Pay **8**
Two Apple Pay Decryption Methods 8
Integration Options 9
Cybersource Decryption 9
Merchant Decryption 9
Unified Checkout 9
Cards Supported for Apple Pay on Visa Platform Connect 10
Payment Services Supported for Apple Pay 11
Summary of Requirements for Using Apple Pay 13
Getting Started with Apple Pay **14**
Requirements for Payment Network Token Support 14
Requirements for Enrolling in the Apple Developer Program 15
Requirements for End-to-End Testing 16
Integrating Apple Pay into Your System **17**
Part 1: Set Up Your Apple Developer Account 18
Starting Enrollment in the Apple Developer Program 18
Completing Enrollment in the Apple Developer Program 18
Registering a New Merchant ID in Your Apple Developer Account 19
Part 2: Create an Apple Pay Payment Processing Certificate 20
Generating a CSR for Cybersource Decryption 20
Generating a CSR for Merchant Decryption 22
Creating a Payment Processing Certificate for Your Merchant ID 22
Part 3: Perform Additional Setup for Apple Pay on the Web 24
Creating an Apple Pay Merchant Identity Certificate 24
Registering Your Merchant Domains with Apple 25
Verifying the Merchant Domains That You Registered with Apple 25
Validating Your Test Integration 26
Integrating Apple Pay into Your Production Environment 26
Processing Apple Pay Transactions **27**
Authorize an Apple Pay Payment with Cybersource Decryption 28

Basic Steps: Authorizing a Payment with Cybersource Decryption....28
Fields Required to Authorize a Payment with Cybersource Decryption....29
Simple Order Example: Authorize a Payment with Cybersource Decryption....30
Authorize an Apple Pay Payment with Merchant Decryption....33
Basic Steps: Authorizing a Payment with Merchant Decryption....33
Fields Required to Authorize a Payment with Merchant Decryption....34
Simple Order Example: Authorize a Payment with Merchant Decryption....35
Process an Apple Pay Sale with Cybersource Decryption....36
Basic Steps: Processing a Sale with Cybersource Decryption....36
Fields Required to Process a Sale with Cybersource Decryption....37
Simple Order Example: Process a Sale with Cybersource Decryption....38
Process an Apple Pay Sale with Merchant Decryption....40
Basic Steps: Processing a Sale with Merchant Decryption....40
Fields Required to Process a Sale with Merchant Decryption....41
Simple Order Example: Process a Sale with Merchant Decryption....42
Reverse an Apple Pay Payment Authorization....43
Basic Steps: Reversing a Payment Authorization....43
Fields Required to Reverse a Payment Authorization....44
Simple Order Example: Reverse a Payment Authorization....44
Capture an Apple Pay Authorization....46
Basic Steps: Capturing an Authorization with Cybersource Decryption....46
Fields Required to Capture a Payment Authorization....47
Simple Order Example: Capture a Payment Authorization....47
Searching for Transactions....49
Searching for Apple Pay Transactions....49
Reference Information....51
Quick Integration for the Cybersource Decryption Method....52
Endpoints for Services Supported with Apple Pay....54
Optional Features Supported with Apple Pay....55
VISA Platform Connect: Specifications and Conditions for Resellers/Partners....56

Apple Pay Developer Guide

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

Audience and Purpose

This guide is for merchants who want to offer Apple Pay in an iOS mobile app or web page and use information from Apple to process payments through Cybersource. The guide describes how to integrate Apple Pay with your system and how to process and search for Apple Pay transactions. Processing is described for Apple Pay payment authorizations, sale (authorization with capture) transactions, authorization reversals, and payment captures. The method you use to extract and decrypt Apple Pay payment data depends on how you integrated Apple Pay into your iOS app or website.

Conventions

These statements appear in this document:

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

Tip: A *Tip* contains information that can help you to complete a tasks or learn a concept.

Related Documentation

Visit the Cybersource documentation hub to find additional technical documentation.

When you perform tasks at the Apple Developer portal, always refer to official Apple documentation for the most up-to-date information:

- Apple ID: https://appleid.apple.com
- Apple Support: https://support.apple.com/
- Apple Developer Support: https://developer.apple.com/support/
- Apple Developer Programs: https://developer.apple.com/programs/
- Apple Developer Help: https://developer.apple.com/help/
- Apple Developer Documentation: https://developer.apple.com/documentation/

Customer Support

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

http://support.visaacceptance.com

Recent Revisions to This Document

25.04.01

This revision contains only editorial changes and no technical updates.

25.01.01

Getting Started with Apple Pay

Added a list of requirements necessary to perform end-to-end testing of Apple Pay transactions. See Requirements for End-to-End Testing (on page 16).

Integrating Apple Pay into Your System

Added more information to the integration section. See these integration tasks:

- Validating Your Test Integration (on page 26)
- Integrating Apple Pay into Your Production Environment (on page 26)

Specifying the Card Type and the Cryptogram Value with Merchant Decryption

Added the **paymentInformation.tokenizedCard.type** field and removed the **paymentInformation.tokenizedCard.cryptogram** field. See these tasks:

- Authorize an Apple Pay Payment with Merchant Decryption (on page 33)
- Process an Apple Pay Sale with Merchant Decryption (on page 40)

Introduction to Apple Pay

Apple Pay is a digital payment solution that enables your customers to make secure and convenient purchases without requiring them to enter their card details or shipping information. You can use the Cybersource platform to process and manage Apple Pay transactions.

When you offer your customers *device apps* enabled for Apple Pay, you can collect payments for purchases made on iPhone and Apple Watch apps. When you offer your customers *Apple Pay on the web*, Apple Pay cardholders can purchase goods and services from within your web app. You can try an Apple Pay test transaction on the Apple Developer site by using the *Apple Pay on the Web Interactive Demo*:

https://applepaydemo.apple.com/

Using Apple Pay on the Cybersource platform can reduce the exposure of sensitive payment data to your system. When a cardholder initiates a purchase from within your Apple Pay enabled app or web page, Apple Pay receives the encrypted transaction. An Apple Pay server returns the transaction payment information re-encrypted with a developer-specific key. The key helps to ensure that only the app or the web page can access the encrypted information.

Customers experience reduced payment friction because their information is tokenized and stored for future use. Customers who configure auto-fill options for their Apple Pay accounts can have payment and card data pre-populate after they sign in to their accounts and authenticate.

Two Apple Pay Decryption Methods

Integration hooks for two Apple Pay decryption methods are built into the Cybersource payment management platform. The two decryption methods—Cybersource decryption and merchant decryption—handle Apple Pay encrypted payment data differently. You will integrate the decryption method that best suits your technical development environment in terms of desired degree of exposure to, or control over, sensitive payment information.

Important: The Apple Pay decryption method that you integrate determines how you will format your API request messages when you authorize a payment or process a sale.

To integrate Apple Pay into your system, you simply register with both Apple and Cybersource, generate keys and certificates (or Cybersource creates and manages them on your behalf), and place the Apple Pay mark on your app or web page. This guide includes instructions for integrating Apple Pay on Cybersource into your system. The instructions cover both of the Apple Pay decryption methods.

Integration Options

Cybersource supports Apple Pay in multiple integration options. Three of the most widely used options are Cybersource decryption, merchant decryption, and Checkout API. Each option presents specific tradeoffs and advantages, and you can select the integration model that best fits your business.

Apple Pay integration is built into the Cybersource payment management platform. Cybersource offers two integration methods for handling the payment data returned by the Apple Pay service for processing payments. In response to an authorization request, Apple Pay returns payment data in an encrypted payload. The encrypted payment data is handled and processed differently, depending on which integration method is used. Both decryption methods support Apple Pay in-app and Apple Pay on the web.

Cybersource Decryption

For Cybersource decryption, you implement Apple Pay directly on your checkout page. You send Cybersource all encrypted payment information that you receive from Apple Pay. Cybersource creates and manages the Apple Pay decryption keys, extracts and decrypts payment information, and maps the information to the appropriate fields for authorization and other payment services on your behalf. Having Cybersource process your Apple Pay transactions reduces your exposure to sensitive payment information.

Merchant Decryption

For merchant decryption, you (the merchant or the integrator) manage all aspects of the Apple Pay implementation, from generation of the payment encryption keys to decryption of the payment response payload from Apple Pay. As a merchant, you submit the Apple Pay payment token and other payment information to Cybersource for processing. With merchant decryption, payment instrument details remain visible to you, and you control the technical development that decrypts this information.

Unified Checkout

Unified Checkout is a consolidated digital acceptance product. Unified Checkout offers a single implementation for multiple payment options. This integration type is designed for merchants looking for a single solution for integrating multiple digital payment options.

Important: Unified Checkout is not covered in this guide. It is mentioned here for the sake of completeness. For information about Unified Checkout, see the *Unified Checkout Integration Guide*.

Cards Supported for Apple Pay on Visa Platform Connect

These cards are supported for Apple Pay on Visa Platform Connect:

- JCB
- mada—debit cards and prepaid cards issued in the Kingdom of Saudi Arabia
- Mastercard—Including mada co-badged cards issued in the Kingdom of Saudi Arabia.
- Visa—Including mada co-badged cards issued in the Kingdom of Saudi Arabia.

Important: 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.

For an overview of financial institutions and payment networks that work together to enable payment services, see the *Payments Developer Guide* . The information includes general descriptions of merchant financial institutions (acquirers), customer financial institutions (issuers), payment networks, and payment processors. The information about processors includes the list of acquirers supported for Visa Platform Connect.

Payment Services Supported for Apple Pay

Apple Pay is supported for the authorization, sale, authorization reversal, and capture services. The credit and void services are also supported for Apple Pay.

Authorization

An authorization confirms that a payment card account holds sufficient funds to pay for a purchase. A successful authorization places a hold on the funds in the account, reducing the cardholder's available limits by the authorized amount. The authorization service is supported with both types of Apple Pay decryption. For more information, see these topics:

- Authorize an Apple Pay Payment with Cybersource Decryption (on page 28)
- Authorize an Apple Pay Payment with Merchant Decryption (on page 33)

Sale (Authorization and Capture)

A sale bundles an authorization and capture into a single transaction. Request the authorization and capture at the same time. Upon a successful transaction, funds are immediately transferred from the cardholder account to the merchant account. The authorization and capture amounts must be the same. The sale service is supported with both types of Apple Pay decryption. For more information, see these topics:

- Process an Apple Pay Sale with Cybersource Decryption (on page 36)
- Process an Apple Pay Sale with Merchant Decryption (on page 40)

Authorization Reversal

Initiate an authorization reversal to reverse an unnecessary or undesired authorization. A successful authorization reversal releases the hold that the authorization placed on the cardholder’s credit card funds. Include in the request message the request ID returned from the previous authorization because the request ID links the reversal to the authorization.

For more information, see Reverse an Apple Pay Payment Authorization (on page 43).

Capture

A capture, also known as settlement, transfers funds from the cardholder’s account to your bank, typically in 2 to 4 days, and it releases the hold that the authorization placed on the cardholder’s credit card funds. Include in the capture request message the request ID returned from the previous authorization because the request ID links the capture to the authorization.

For more information, see Capture an Apple Pay Authorization (on page 46).

Credit

A is a payment refund from your bank to the cardholder for a payment that has already been captured. To initiate a , send a request message to the **credit service** and include the request ID that was returned in the response to the capture request. Because the request ID links to the cardholder's billing and account information, you are not required to include those fields in the credit request. Unless otherwise specified, you must request a follow-on credit within 180 days of a settlement.

For more information about standalone credit requests and credit authorization results, see the "Standard Payments Processing" section of the *Payments Developer Guide*.

Void

You can void an Apple Pay capture or credit that was submitted but is not yet processed by the processor. You send a request for a capture void and a credit void to the same endpoint, but different fields are required for the two types of voids. A void is linked to a capture or credit transaction through the request ID of the transaction you want to void.

For more information about void requests and responses, see the "Standard Payments Processing" section of the *Payments Developer Guide*.

Summary of Requirements for Using Apple Pay

This topic lists the key requirements for using Apple Pay. More detailed information is provided in Getting Started with Apple Pay (on page 14).

A Cybersource merchant account.

If you do not have a merchant account, contact your Cybersource sales representative.

Apple Pay enabled for your Cybersource account.

If Apple Pay is not enabled, contact your Cybersource representative.

A Cybersource Business Center sandbox account.

To create a test account, visit the Business Center Sandbox Account Sign-Up page.

Test account login page: `https://businesscentertest.cybersource.com/ebc2`

A Cybersource Business Center production account with a supported processor.

If you do not have a production account, contact your Cybersource sales representative.

Production account login page: `https://businesscenter.cybersource.com`

Production account login in India: `https://businesscenter.in.cybersource.in.com`

Important:
Apple Pay relies on authorizations with payment network tokens. Your environment must meet these requirements in order to support payment network tokenization:

- Your processor supports payment network tokens.
- Cybersource supports payment network tokens with your processor.

If your environment does not meet both requirements, you have these options:

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

Getting Started with Apple Pay

This section describes requirements for integrating Apple Pay into your system:

- Requirements for payment network token support
- Requirements for enrolling in the Apple Developer Program
- Requirements for end-to-end testing

Some of the requirements will be met as you complete the steps described in Integrating Apple Pay into Your System (on page 17).

Requirements for Payment Network Token Support

Apple Pay relies on authorizations being processed with *payment network tokens*. Make sure your processor and your Cybersource test accounts and production accounts meet these requirements.

Your processor supports payment network tokens and Cybersource supports payment network tokens with your processor.

If your processor does not support payment network tokens, or if Cybersource does not support payment network tokens with your processor, you must obtain a new Cybersource merchant account with a processor that supports payment network tokens.

Your Cybersource test account supports payment network tokens with your payment processor.

In order to configure and validate a test integration of Apple Pay, your Cybersource merchant test account supports payment network tokens with your processor.

If you do not have a merchant test account that meets this criteria, contact your Cybersource sales representative.

Your Cybersource production account supports payment network tokens with your payment processor.

In order to use your Apple Pay implementation in a production environment, your Cybersource merchant production account supports network tokens with your processor.

If you do not have a production account that meets these criteria, contact your Cybersource sales representative.

Requirements for Enrolling in the Apple Developer Program

The Apple Pay integration process requires you to enroll your organization in the Apple Developer Program. The exact requirements can vary depending on the specifics of your app and business. Always refer to official Apple documentation for the most up-to-date information.

Important: If you are enabling Apple Pay digital payments on Unified Checkout, see the *Unified Checkout Integration Guide*. The requirements described in this topic apply only if you are integrating *standalone Apple Pay* for Cybersource decryption or merchant decryption.

Your Apple device runs the latest Apple Developer app.

The machine that you use for **enrollment** must be an Apple device that run the latest version of the Apple Developer app. You can download the app from the Apple App Store.

You have an Apple ID with two-factor authentication and a valid payment method.

An Apple ID grants access to Apple Developer resources, including documentation, sample code, forums, and technical support. This Apple ID can be different from the Apple ID that you used to sign in to your enrollment device.

Two-factor authentication is required in order to sign in to your Apple Developer account and manage your account in the **Certificates, Identifiers & Profiles** page. The payment method associated with your Apple ID is used to pay the annual subscription fee.

You will be the Account Holder or Admin for your organization.

As the person enrolling your organization in the Apple Developer Program, you automatically become the Account Holder for your project. Either of these roles enables you to generate an Apple Pay encryption key to encrypt payment data during the payment flow.

Each developer in your organization must be assigned to your project as an Account Holder or an Admin. Only an Account Holder or an Admin can create developer accounts for their team.

You have the information about your organizations that is required for enrollment.

You must provide basic business details and a D-U-N-S Number. Apple verifies that the information you provide is accurate and current before approving your enrollment.

You have the authorization to pay the annual membership fee.

The annual membership fee that you pay to Apple enables you to distribute your apps on the Apple App Store and to access beta software, beta testing tools, advanced app capabilities, and app analytics.

You must accept the terms of the license agreement.

You must agree to the Apple Developer Program License Agreement.

Requirements for End-to-End Testing

This topic lists the requirements for supporting end-to-end testing of Apple Pay transaction processing in a testing environment. End-to-end testing requires that test payment cards have been loaded to your Apple Developer sandbox tester account wallet.

Important: If these requirements are not met, Apple Pay servers will not return a valid payload in your test environment.

You have an Apple sandbox tester account.

For sandbox setup instructions, see the *Sandbox Testing* page in the Apple Developer portal: https://developer.apple.com/apple-pay/sandbox-testing/

Follow the steps described in the section *Create a Sandbox Tester Account*.

If you are integrating the merchant decryption model, you have an Apple test device.

You must register the device with Apple using your Apple Developer account. You will use test device to create the certificate signing request (CSR) that you will use to associate your Apple sandbox tester account with your test environment.

If you are integrating the merchant decryption model, you will use the Apple device in one of the tasks described in Part 2: Create an Apple Pay Payment Processing Certificate (on page 20).

You have a test environment that you can access by logging in to your Apple sandbox tester account.

To create this access, you will use the sandbox tester account to create an Apple test merchant ID. You will use the test ID to create a CSR and use the CSR to create an Apple Pay payment processing certificate. Apple Pay servers use this certificate to encrypt payment data using a key known to your test environment.

The steps for associating a sandbox tester account with your test environment (as well as for associating an Apple production account with your production environment) are included in Part 2: Create an Apple Pay Payment Processing Certificate (on page 20).

You have test payment cards in the wallet of your Apple sandbox tester account.

You will be instructed to add test cards to your sandbox tester account in Validating Your Test Integration (on page 26).

Integrating Apple Pay into Your System

This section describes how to integrate Apple Pay into your iOS app or website. The integration tasks are organized into three parts. The second part provides separate steps for the two different decryption models. The third part applies only if you will be supporting Apple Pay on the web.

You will perform the integration tasks twice: First in your test environment and, after you validate your test integration, a second time in your production environment.

- **Part 1: Set Up Your Apple Developer Account**. You will enroll your organization in the Apple Developer Program, create an *Apple merchant ID*, and register it in your developer account.
- **Part 2: Create an Apple Pay Payment Processing Certificate**. This certificate is associated with your merchant ID, and it is used by Apple Pay servers to encrypt payment data.
    - You will generate a *certificate signing request* (CSR) at the system that will handle Apple Pay payload decryption. For Cybersource decryption, you will generate the CSR at the Cybersource Business Center user interface. For merchant decryption, you will generate the CSR at your Apple device.
    - You will upload the CSR with the public key to your Apple Developer account and use the CSR to create a *payments processing certificate* for your merchant ID and Apple Pay.
- **Part 3: Perform Additional Setup for Apple Pay on the Web**. If you offer your customers Apple Pay on the web, you will create an *Apple Pay merchant identity certificate*, associate the certificate with your merchant ID, and register *each merchant domain* that will process Apple Pay transactions.

Tip: If you are integrating **Apple Pay with Cybersource decryption** and you are experienced in creating Apple Pay payment processing certificates, you can use the Quick Integration for the Cybersource Decryption Method (on page 52) instead of the detailed steps in this section.

Part 1: Set Up Your Apple Developer Account

Complete the tasks in this section to enroll your organization in the Apple Developer program and register a new Apple merchant ID.

Starting Enrollment in the Apple Developer Program

Enrolling in the Apple Developer program as an organization enables you to associate multiple developer accounts with your Apple Developer account. Multiple developer accounts can be beneficial if you have a large project with a team of developers.

For the first phase of the enrollment process, you log in to your Apple Developer account and submit information about your organization to Apple.

Important: When you perform tasks at the Apple Developer portal, always refer to official Apple documentation for the most up-to-date information.

Follow these steps to start the enrollment process:

1. Launch the Apple Developer app on your device.
2. Click **Account**, and sign in with your Apple ID.
3. If prompted, review the Apple Developer Agreement and click **Agree**.
4. Click **Enroll Now**, review the program benefits and requirements, and then click **Continue**.
5. At the prompts, enter your information as the Account Holder.
6. At the prompts, enter information about your organization.

After Apple verifies your information and approves your enrollment, it sends you an email that describes the next steps.

Completing Enrollment in the Apple Developer Program

When you receive your approval email from Apple, you will log in to your Apple Developer account again and complete the enrollment process.

Important: When you perform tasks at the Apple Developer portal, always refer to official Apple documentation for the most up-to-date information.

Follow these steps to complete the enrollment process:

1. Launch the Apple Developer app on the device you used to start the enrollment process.
2. Click **Account** and sign in with the Apple ID you used to start the enrollment process.
3. Click **Continue Your Enrollment**, review the terms of the Apple Developer Program License Agreement, and then click **Agree**.
4. Review the annual membership subscription details and click **Subscribe**.

Registering a New Merchant ID in Your Apple Developer Account

Finish setting up your Apple Developer account by creating and registering a merchant ID for each environment. A registered merchant ID uniquely identifies you to Apple Pay as a valid entity that can accept payments.

In order to support multiple environments, such as sandbox and production, you can create multiple merchant IDs in your Apple Developer account.

Important: When you perform tasks at the Apple Developer portal, always refer to official Apple documentation for the most up-to-date information.

Follow these steps to create a merchant ID and to register it in your Apple Developer account:

1. Log in to your Apple Developer account.
2. In the left navigation panel, select **Certificates, Identifiers & Profiles**.
3. Click **Identifiers**.
4. Click the plus sign (**+**) on the top left.
5. Select **Merchant IDs** and click **Continue**.
6. Enter a merchant description and identifier name.
7. Click **Continue**.
8. Verify that you entered the merchant information correctly.
9. Click **Register**.

Part 2: Create an Apple Pay Payment Processing Certificate

Complete the tasks in this section to create an Apple Pay payment processing certificate. Apple Pay servers use this certificate to encrypt payment data. Creation of an Apple Pay payment processing certificate consists of two tasks:

- Generating a certificate signing request (CSR).
- Using the CSR to create an Apple Pay payment processing certificate.

Important:
When you generate a CSR, the sequence of steps you will perform depends on whether you are integrating Cybersource decryption and merchant decryption.

- If you are integrating Cybersource decryption, you will generate a CSR at the Cybersource Business Center. See Generating a CSR for Cybersource Decryption (on page 20).
- If you are integrating merchant decryption, you will generate a CSR at your Apple device. See Generating a CSR for Merchant Decryption (on page 22).

Generating a CSR for Cybersource Decryption

Important: These steps apply to setting up Cybersource decryption only. If you are integrating the merchant decryption model of Apple Pay into your system, follow the steps in Generating a CSR for Merchant Decryption (on page 22) instead.

For Cybersource decryption, you will use your Cybersource account in the Business Center to generate a certificate signing request (CSR). You will use the Apple Pay Registration page within the Business Center.

If you do not have an Admin account or an account with write access, contact your Account Admin, Cybersource sales engineer, alliance partner, or technical account manager.

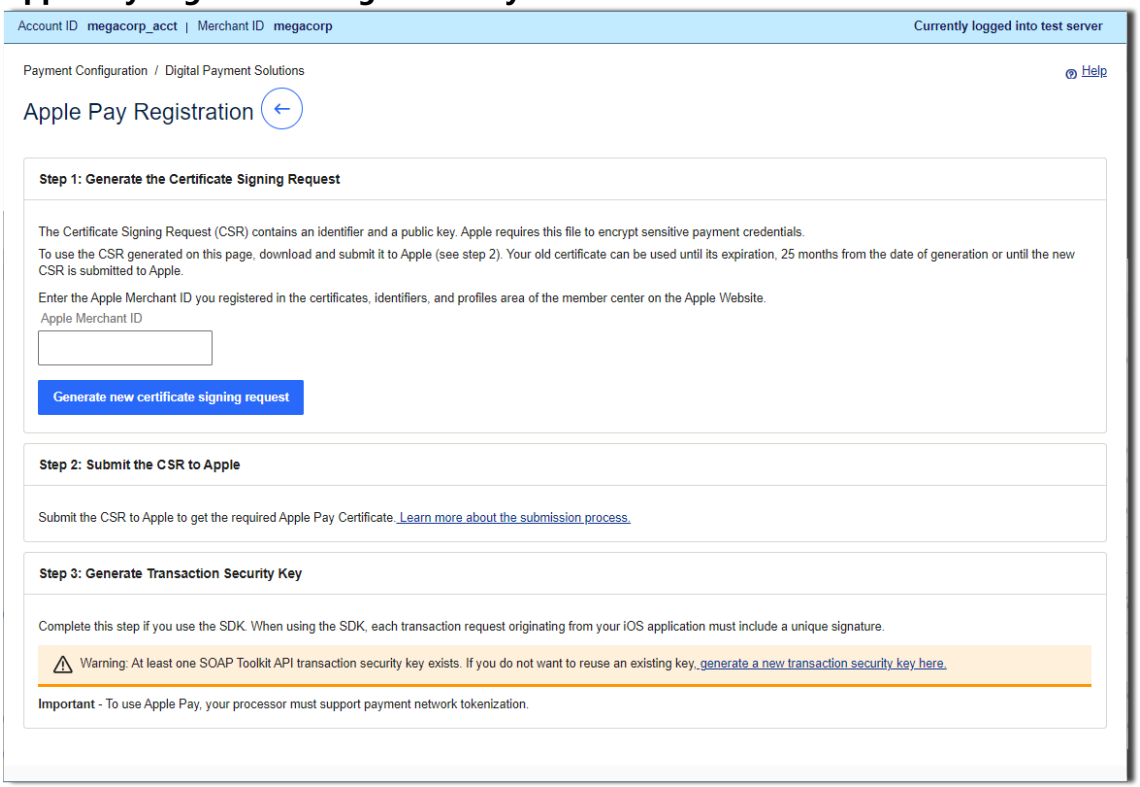
Follow these steps at the Cybersource Business Center to generate a CSR:

1. Log in to your Cybersource merchant account in the Business Center.
   Production: `https://businesscenter.cybersource.com`
   Production in India: `https://businesscenter.in.cybersource.in.com`
   Test: `https://businesscentertest.cybersource.com/ebc2`
2. In the left navigation panel, select **Payment Configuration**.

3. Choose **Digital Payment Solutions**.
   The Digital Payment Solutions page appears.

4. Click **Configure** for Apple Pay.
   The Apple Pay Registration page appears.
   This image shows the Apple Pay Registration page in the Cybersource Business Center.

Apple Pay Registration Page in the Cybersource Business Center Interface



5. Enter the Apple merchant ID that you created and registered in your Apple Developer account. These steps are described in Registering a New Merchant ID in Your Apple Developer Account (on page 19). This value should match the id you set up with Apple including the "merchant" prefix.

6. Click **Generate new certificate signing request**.

7. Click the download icon next to the key.

8. Download the certificate request file (a file with a *.certSigningRequest* file extension) to your local machine.

9. Use your browser controls to save the file to your local machine.
   In the next task, you will upload the CSR file to your Apple Developer account.

10. Proceed to Creating a Payment Processing Certificate for Your Merchant ID (on page 22).

Generating a CSR for Merchant Decryption

Important: These steps apply to setting up merchant decryption only. If you are integrating the Cybersource decryption model of Apple Pay into your system, follow the steps in Generating a CSR for Cybersource Decryption (on page 20) instead.

For merchant decryption, you will use your Apple device to generate a certificate signing request (CSR).

Important: When you perform tasks at the Apple Developer portal, always refer to official Apple documentation for the most up-to-date information.

Follow these steps at your Apple device to generate a CSR:

1. Sign in to your Apple Developer account as the Account Holder or as an Admin and select **Certificates, Identifiers & Profiles**.
2. Click **Identifiers** in the sidebar.
3. Select **Merchant IDs** and click **Continue**.
4. Under Identifiers, select **Merchant IDs** using the filter on the top right.
5. On the right, select your merchant identifier.
6. Under Apple Pay Payment Processing Certificate, click **Create Certificate**.
7. Proceed to Creating a Payment Processing Certificate for Your Merchant ID (on page 22).

Creating a Payment Processing Certificate for Your Merchant ID

Using the certificate signing request that you just created, you will create an Apple payment processing certificate and associate the certificate with your Apple merchant ID that you created before that. Those earlier tasks are described in these topics:

- Registering a New Merchant ID in Your Apple Developer Account (on page 19)
- Creating a Payment Processing Certificate for Your Merchant ID (on page 22)

Apple Pay uses the payment processing certificate to encrypt the customer's payment information. This certificate expires every 25 months. If the certificate expires or is revoked, you can recreate it.

Important: When you perform tasks at the Apple Developer portal, always refer to official Apple documentation for the most up-to-date information.

Follow these steps to create a payment processing certificate for your Apple Pay merchant ID:

1. Sign in to your Apple Developer account as the Account Holder or as an Admin and select **Certificates, Identifiers & Profiles**.

2. Upload the CSR file and associate the CSR with your merchant ID.
   The CSR contains your Apple merchant ID and a public key that Apple Pay uses to encrypt sensitive payment data.

   a. Click **Identifiers** in the sidebar.

   b. Select **Merchant IDs** using the filter on the top right.

   c. On the right, select your merchant ID.
   If a banner at the top of the page prompts you, you need to accept an agreement. Click **Review Agreement** and follow the instructions that appear.

   d. Under Apple Pay Payment Processing Certificate, click **Create Certificate**.

3. Create a payment processing certificate and download the certificate to your local machine.

   a. Click **Choose File** and select the CSR file that you uploaded.
   The CSR file has the filename extension *.certSigningRequest*.

   b. Click **Choose**.

   c. Click **Continue**.

   d. Click **Download**.
   The payment processing certificate (a file with the filename extension *.cert*) appears in your Downloads folder.

4. Go to the next task.

   ◦ If you offer your customers Apple Pay in a web page, go to Part 3: Perform Additional Setup for Apple Pay on the Web (on page 24).

   ◦ Otherwise, proceed to Validating Your Test Integration (on page 26).

Part 3: Perform Additional Setup for Apple Pay on the Web

If you develop web pages that support Apple Pay on the Web, your customers can use Apple Pay to purchase goods and services from within your web page. You can use the same Apple Pay merchant ID and Apple Pay payment processing certificate as required for Apple Pay in-app implementations. However, Apple Pay on the Web requires additional set-up tasks that you perform in your Apple Developer account:

- Creating an Apple Pay merchant identity certificate
- Registering your merchant domains with Apple

If you created multiple merchant ID and payment processing certificate pairs to support multiple environments, such as sandbox and production, you must associate each ID-and-certificate pair with a unique merchant identify certificate.

Creating an Apple Pay Merchant Identity Certificate

If you offer your customers Apple Pay in a web page, you must create an Apple Pay merchant identity certificate and associate it with your merchant ID. You need this Transport Layer Security (TLS) certificate in order to authenticate your sessions with the Apple Pay servers.

Important: When you perform tasks at the Apple Developer portal, always refer to official Apple documentation for the most up-to-date information.

Follow these steps to create an Apple Pay merchant identity certificate:

1. Log in to your Apple Developer merchant account as an Account Holder or Admin.
2. In the left navigation panel, select **Certificates, Identifiers & Profiles**.
3. Perform these steps for each merchant identity certificate you need to create:
    a. Click **Identifiers**, and click the plus sign (**+**) on the top left.
    b. Select **Merchant IDs** and click **Continue**.
    c. Enter the merchant description and identifier name, and then click **Continue**.
    d. Click **Register**.

Registering Your Merchant Domains with Apple

Each merchant domain in your organization that will process Apple Pay transactions must be registered with Apple.

Important: When you perform tasks at the Apple Developer portal, always refer to official Apple documentation for the most up-to-date information.

Follow these steps to register your merchant domains with Apple:

1. Log in to your Apple Developer merchant account as an Account Holder or Admin.
2. In the left navigation panel, select **Certificates, Identifiers & Profiles**.
3. Perform these steps for each merchant domain that you registered with Apple:
    a. Click **Identifiers**, and select **Merchant IDs** in the pop-up menu on the top right.
    b. On the right, select your merchant identifier.
    c. Under Merchant Domains, click **Add Domain**. Enter the fully qualified name of the domain and click **Save**.
    d. Click **Download**, place the downloaded file in the specified locations, and click **Verify**.
4. After you add all merchant domains that will process Apple Pay transactions, click **Done**.

Verifying the Merchant Domains That You Registered with Apple

Follow these steps to verify the merchant domains you registered with Apple:

1. Log in to your Apple Developer merchant account as an Account Holder or Admin.
2. In the left navigation panel, select **Certificates, Identifiers & Profiles**.
3. Perform these steps for each merchant domain that you registered with Apple:
    a. Click **Identifiers**, and select **Merchant IDs** in the pop-up menu on the top right.
    b. On the right, select your merchant identifier.
    c. Under Merchant Domains, click **Verify** next to the domain name.
    d. Follow the instructions that appear on the screen.

You can now proceed to Validating Your Test Integration (on page 26).

Validating Your Test Integration

Before you integrate Apple Pay into your production environment, validate your test integration of Apple Pay.

Follow these steps to validate the integration in your test environment:

1. Make sure your system is prepared for end-to-end testing.
   See Requirements for End-to-End Testing (on page 16).

2. Add test payment cards to the wallet of your Apple sandbox tester account.
   Instructions are provided in the *Sandbox Testing* page on the Apple Developer portal:

   https://developer.apple.com/apple-pay/sandbox-testing/

   a. Follow the steps in the *Create a Sandbox Tester Account* section.
      Make sure the user account has permissions to use Apple Pay. You will use this account to log in to devices and services.

   b. Follow the steps in the *Adding a Test Card Number* section.

3. Using the Simple Order API, send Apple Pay transaction requests to the test endpoints.
   Refer to the tasks in Processing Apple Pay Transactions (on page 27).

4. Adjust your integration settings as needed until your test transactions complete successfully.

You can now proceed to Integrating Apple Pay into Your Production Environment (on page 26).

Integrating Apple Pay into Your Production Environment

After you validate Apple Pay in your test environment, you can integrate Apple Pay into your production environment.

Follow these steps to integrate Apple Pay into your production environment:

1. Use your Apple merchant ID to generate a certificate signing request (CSR) and create a *production* Apple Pay payment processing certificate.
   See Part 2: Create an Apple Pay Payment Processing Certificate (on page 20).

2. If you offer your customers Apple Pay on the Web, perform the additional setup steps for your production environment.
   See Part 3: Perform Additional Setup for Apple Pay on the Web (on page 24).

You can now proceed to Processing Apple Pay Transactions (on page 27).

Processing Apple Pay Transactions

Cybersource supports these payment services for Apple Pay: Authorization, Sale, Authorization Reversal, and Capture. This section of the guide contains tasks that show you how to use those services to process Apple Pay transactions:

- Authorize an Apple Pay payment with Cybersource decryption
- Authorize an Apple Pay payment with merchant decryption
- Process an Apple Pay sale with Cybersource decryption
- Process an Apple Pay sale with merchant decryption
- Reverse the authorization of an Apple Pay payment
- Capture an authorized Apple Pay payment

For each task in this section, the example request message includes the **paymentSolution** Simple Order API field set to `001`. This value identifies Apple Pay as the digital payment solution.

Important: In response to your successful payment authorization request, Apple Pay returns an encrypted payload that contains sensitive payment information. The method you use to extract and decrypt the payment data depends on how you integrated Apple Pay into your system.

For the list of supported cards, see Cards Supported for Apple Pay on Visa Platform Connect (on page 10).

For high-level descriptions of the Cybersource decryption method and the merchant decryption method, see Integration Options (on page 9).

Authorize an Apple Pay Payment with Cybersource Decryption

The topics in this section shows you how to authorize an Apple Pay payment transaction with the *Cybersource decryption* implementation of Apple Pay.

- Basic Steps: Authorizing a Payment with Cybersource Decryption (on page 28)
- Fields Required to Authorize a Payment with Cybersource Decryption (on page 29)
- Simple Order Example: Authorize a Payment with Cybersource Decryption (on page 30)

Important: In the example, the payment being authorized was made using a **Visa card**, and the processor is **Visa Platform Connect**. If you want to replicate this example for a different card or a different processor, you will need to change some of the API fields in the request message.

For more information, see Cards Supported for Apple Pay on Visa Platform Connect (on page 10).

For general information about basic authorizations, see the "Standard Payments Processing" section of the *Payments Developer Guide*.

Basic Steps: Authorizing a Payment with Cybersource Decryption

Follow these steps to request an Apple Pay payment authorization with Cybersource decryption:

1. Create the request message with the required Simple Order API fields.
    - Use the API fields listed in Fields Required to Authorize a Payment with Cybersource Decryption (on page 29).
    - Refer to the example in Simple Order Example: Authorize a Payment with Cybersource Decryption (on page 30).
2. Set the **ccAuthService_run** field to `true`.
3. Send the request to `https://ics2ws.ic3.com/commerce/1.x/transactionProcessor`.
4. Verify the response messages to make sure that the request was successful. A value of `ACCEPT` for the decision field indicates success.

Fields Required to Authorize a Payment with Cybersource Decryption

As a best practice, include these Simple Order fields in your request for an authorization transaction with the Cybersource decryption implementation of Apple Pay.

Depending on your processor, your geographic location, and whether the relaxed address verification system (RAVS) is enabled for your account, some of these fields might not be required. It is your responsibility to determine whether an API field can be omitted from the transaction you are requesting. For information about the relaxed requirements for address data and expiration dates in payment transactions, see the *Payments Developer Guide*.

billTo_city

billTo_country

billTo_email

billTo_firstName

billTo_lastName

billTo_postalCode

billTo_state

billTo_street1

ccAuthService_overridePaymentMethod

> **Important:**
> This field is available only for authorizations on the Cielo3 processor.
> **[[[VERIFY]]]** Possible values:
>
> - `CR`: Credit card
> - `DB`: Debit card

Obtain this information from the card_usage field [[[in the cardholder provides this information during the payment process]]].

encryptedPayment_data

Set the value to the encrypted payment data value returned by the Full Wallet request.

encryptedPayment_descriptor

Set the value to `RklEPUNPTU1PTi5BUFBMRS5JTkFQUC5QQVlNRU5U`, the format of the encrypted payment data for Apple Pay.

encryptedPayment_encoding

Set the value to `Base64`, the encoding method for Apple Pay transactions.

merchantReferenceCode

paymentSolution

Set the value to `001` to identify Apple Pay as the digital payment solution.

purchaseTotals_currency

purchaseTotals_grandTotalAmount

Simple Order Example: Authorize a Payment with Cybersource Decryption

Request

```
ccAuthService_run=true
merchantReferenceCode=TC_1231223
paymentSolution=001
purchaseTotals_grandTotalAmount=10
purchaseTotals_currency=GBP
billTo_firstName=John
billTo_lastName=Doe
billTo_street1=901 Metro Center Blvd
billTo_city=Foster City
billTo_state=CA
billTo_postalCode=94404
billTo_country=US
billTo_email=test@cybersource.com
encryptedPayment_data=eyJkYXRhIjoiSnN0R3lkVWsyRG1mWm9YMUxNWUVoVXNicFJvTTRJMGVGK3Vn
bkFjUmEwczNnS0xicl
c0MVFWb1BhWEtWXC9BSU0wNWlRNitEcm1UVXUyTnVQRE91UkxIeHpkOHhqUXpkeHhQYXdMMURcL1FEa2pm
akJvY2d6
OEc5UG1KanpiY1ZlZm5EWmd6UjlaNExlT3NJVlAzaG9HcEVMYWVnV3pTRXhBNEtIcDlmODdESm11WXRGVE
xIbW5qSH
Z0UytFa1FuWU00OGJLbnN4NGR1dm5RUVordFVwak9vOFEyZEplQ2s5WWFsSmY3NXVQaEtIVkRQVmg2dE12
WExuWmpn
YjYyZXlzNytOdGlXK1dRWHJydnJ0TmxVVFwvSDYyR2x6VGxFeHBTK2FGdWVqZ3k3d2FjK3B0UERWWHR1XC
9IbGdPck
5aTThcL0o4SFhCZ04yOGd5MEJuVzYxcnc1WlJ1ZWxFODNWc1FcLzlMUmVHczF0anA2aE0wZ01cLzJhaTFo
d2pkQzcy
VmtWNmxTT0FzZ0s5SVB0bHpPTm4rekd2aEwwTER3K01temV3VFwvNWczT3FGeHFtbURHODFqTTJ1TFlcLz
d2WURlXC
9nY2lPXC9najFwa2xoTmJsXC9RPT0iLCJ2ZXJzaW9uIjoiRUNfdjEiLCJoZWFkZXIiOnsiYXBwbGljYXRp
b25EYXRh
```

```
IjoiNDE3MDcwNkM2OTYzNjE3NDY5NkY2RTQ0NjE3NDYxIiwidHJhbnNhY3Rpb25JZCI6IjU0NzI2MTZFNz
M2MTYzNz
Q2OTZGNkU0OTQ0IiwiZXBoZW1lcmFsUHVibGljS2V5IjoiTUlJQlN6Q0NBUU1HQnlxR1NNNDlBZ0V3Z2Zj
Q0FRRXdM
QVlIS29aSXpqMEJBUUloQVBcL1wvXC9cLzhBQUFBQkFBQUFBQUFBQUFBQUFcL1wvXC9cL1wvXC9cL1
wvXC9cL1
wvXC9cL1wvXC9cL01Gc0VJUFwvXC9cL1wvOEFBQUFCQUFBQUFBQUFBQUFBQVwvXC9cL1wvXC9cL1wv
XC9cL1wv
XC9cL1wvXC9cLzhCQ0JheGpYWXFqcVQ1N1BydlZWMm1JYThaUjBHc014VHNQWTd6ancrSjlKZ1N3TVZBTV
NkTmdpRz
V3U1RhbVo0NFJPZEpyZUJuMzZRQkVFRWF4ZlI4dUVzUWtmNHZPYmxZNlJBOG5jRGZZRXQ2ek9nOUtFNVJk
aVl3cFpQ
NDBMaVwvaHBcL200N242MHA4RDU0V0s4NHpWMnN4WHM3THRrQm9ONzlSOVFJaEFQXC9cL1wvXC84QUFBQU
FcL1wvXC
9cL1wvXC9cL1wvXC9cLys4NXZxdHB4ZWVoUE81eXNMOFl5VlJBZ0VCQTBJQUJENm5QV05Ta1dROVpSa2VP
b3NJV2c1
RFF6S3JoQ0ZIN1NsOHB6Z3VKZFR0OHU0N0o5aW42UnZTVGJGTHdXWXQrd2pFRkh2ZWFtaHlKeDlZSXNKTU
lJQT0iLC
JwdWJsaWNLZXlIYXNoIjoiVkFXMVlEQXRFS3dVcWpZbEtOVDNZeFlHYysxSXZ3M1RrWm9kNW9NSGd2TT0i
fSwic2ln
bmF0dXJlIjoiTUlJRFFnWUpLb1pJaHZjTkFRY0NvSUlETXpDQ0F5OENBUUV4Q3pBSkJnVXJEZ01DR2dVQU
1Bc0dDU3
FHU0liM0RRRUhBYUNDQWlzd2dnSW5NSUlCbEtBREFnRUNBaEJjbCtQZjMrVTRwazEzblZEOW53UVFNQWtH
QlNzT0F3
SWRCUUF3SnpFbE1DTUdBMVVFQXg0Y0FHTUFhQUJ0QUdFQWFRQkFBSFlBYVFCekFHRUFMZ0JqQUc4QWJUQW
VGdzB4Tk
RBeE1ERXdOakF3TURCYUZ3MHlOREF4TURFd05qQXdNREJhTUNjeEpUQWpCZ05WQkFNZUhBQmpBR2dBYlFC
aEFHa0FR
QUIyQUdrQWN3QmhBQzRBWXdCdkFHMHdnWjh3RFFZSktvWklodmNOQVFFQkJRQURnWTBBTUlHSkFvR0JBTk
M4K2tndG
dtdldGMU96amdETnJqVEVCUnVvXC81TUt2bE0xNDZwQWY3R3g0MWJsRTl3NGZJWEpBRDdGZk83UUtqSVhZ
TnQzOXJM
eXk3eER3YlwvNUlrWk02MFRaMmlJMXBqNTVVYzhmZDRmek9wazNmdFphUUdYTkxZcHRHMWQ5VjdJUzgyT3
VwOU1Nbz
FCUFZyWFRQSE5jc005OUVQVW5QcWRiZUdjODdtMHJBZ01CQUFHalhEQmFNRmdHQTFVZEFRUlJNRStBRUha
V1ByV3RK
ZDdZWjQzMWhDZzdZRlNoS1RBbk1TVXdJd1lEVlFRREhod0FZd0JvQUcwQVlRQnBBRUFBZGdCcEFITUFZUU
F1QUdNQW
J3QnRnaEJjbCtQZjMrVTRwazEzblZEOW53UVFNQWtHQlNzT0F3SWRCUUFEZ1lFQWJVS1lDa3VJS1M5UVEy
bUZjTVlS
RUltMmwrWGc4XC9KWHYrR0JWUUprT0tvc2NZNGlOREZBXC9iUWxvZ2Y5TExVODRUSHdOUm5zdlYzUHJ2N1
JUWTgxZ3
EwZHRDOHpZY0FhQWtDSElJM3lxTW5KNEFPdTZFT1c5a0prMjMyZ1NFN1dsQ3RIYmZMU0tmdVNnUVg4S1hR
WXVaTGsy
UnI2M044QXBYc1h3QkwzY0oweGdlQXdnZDBDQVFFd096QW5NU1V3SXdZRFZRUURIaHdBWXdCb0FHMEFZUU
JwQUVBQW
RnQnBBSE1BWVFBdUFHTUFid0J0QWhCY2wrUGYzK1U0cGsxM25WRDlud1FRTUFrR0JTc09Bd0lhQlFBd0RR
WUpLb1pJ
```

```
aHZjTkFRRUJCUUFFZ1lBVVY2amlDOWVFOHdaQUhRRjJoZzB1aXBJdm9BWWdycmN5YlwvVUc3R2dhbWliczJQcEtpSW
JnSFZ4THJMNFBxU0FtSllHMHFMcFA0dU84Nloxb1VpNUdnK2xaNHZKM0NZRUt5bVpjanFGT3QwZkg0a2lPSStlRFRO
SFhac05uV0lCZnkzV3Uwdlk3UUpXMzN6M1Y3alZWdzlqYlFEdllzbXBIelAxWmp2ZlhhUT09In0=
encryptedPayment_descriptor=RklEPUNPTU1PTi5BUFBMRS5JTkFQUC5QQVlNRU5U
encryptedPayment_encoding=Base64
```

Response to a Successful Request

```
{
}
```

Authorize an Apple Pay Payment with Merchant Decryption

The topics in this section shows you how to authorize an Apple Pay payment transaction with the *merchant decryption* implementation of Apple Pay.

- Basic Steps: Authorizing a Payment with Merchant Decryption (on page 33)
- Fields Required to Authorize a Payment with Merchant Decryption (on page 34)
- Simple Order Example: Authorize a Payment with Merchant Decryption (on page 35)

Important: In the example, the payment being authorized was made using a **Visa card**, and the processor is **Visa Platform Connect**. If you want to replicate this example for a different card or a different processor, you will need to change some of the API fields in the request message.

For more information, see Cards Supported for Apple Pay on Visa Platform Connect (on page 10).

For general information about basic authorizations, see the "Standard Payments Processing" section of the *Payments Developer Guide*.

Basic Steps: Authorizing a Payment with Merchant Decryption

Follow these steps to request an Apple Pay payment authorization with merchant decryption:

1. Create the request message with the required Simple Order API fields.
    - Use the API fields listed in Fields Required to Authorize a Payment with Merchant Decryption (on page 34).
    - Refer to the example in Simple Order Example: Authorize a Payment with Merchant Decryption (on page 35).
2. Set the **ccAuthService_run** field to `true`.
3. Send the request to `https://ics2ws.ic3.com/commerce/1.x/transactionProcessor`.
4. Verify the response messages to make sure that the request was successful. A value of `ACCEPT` for the decision field indicates success.

Fields Required to Authorize a Payment with Merchant Decryption

As a best practice, include these Simple Order fields in your request for an authorization transaction with the merchant decryption implementation of Apple Pay.

Depending on your processor, your geographic location, and whether the relaxed address verification system (RAVS) is enabled for your account, some of these fields might not be required. It is your responsibility to determine whether an API field can be omitted from the transaction you are requesting. For information about the relaxed requirements for address data and expiration dates in payment transactions, see the *Payments Developer Guide*.

billTo_city

billTo_country

billTo_email

billTo_firstName

billTo_lastName

billTo_postalCode

billTo_state

billTo_street1

card_accountNumber

Set the value to the customer's payment network token value.

ccAuthService_commerceIndicator

Set the value to the indicator that corresponds to the ecommerce value extracted from the Apple Pay response payload:

- `aesk`: American Express
- `dipg`: Discover
- `js:` JCB
- `spa`: Mastercard
- `vbv`: Visa

ccAuthService_networkTokenCryptogram

Set this field to the network token cryptogram.

merchantReferenceCode

paymentNetworkToken_transactionType

Set the value to `1` to identify the payload as a network token delivered from a customer's wallet.

paymentSolution

Set the value to `001` to identify Apple Pay as the digital payment solution.

purchaseTotals_currency

purchaseTotals_grandTotalAmount

token_expirationMonth

token_expirationYear

Simple Order Example: Authorize a Payment with Merchant Decryption

Request

```
ccAuthService_run=true
merchantReferenceCode=TC_1231223
paymentSolution=001
ccAuthService_commerceIndicator=vbv
card_accountNumber=4111111111111111
token_expirationMonth=12
token_expirationYear=2031
ccAuthService_networkTokenCryptogram=AQAAAAAACyWowDMAAAAAgQVgAgA=
paymentNetworkToken_transactionType=1
purchaseTotals_grandTotalAmount=10
purchaseTotals_currency=USD
billTo_firstName=John
billTo_lastName=Doe
billTo_street1=901 Metro Center Blvd
billTo_city=Foster City
billTo_state=CA
billTo_postalCode=94404
billTo_country=US
billTo_email=test@cybersource.com
```

Response to a Successful Request

```
{
}
```

Process an Apple Pay Sale with Cybersource Decryption

The topics in this section shows you how to process an Apple Pay sale with the *Cybersource decryption* implementation of Apple Pay.

- Basic Steps: Processing a Sale with Cybersource Decryption (on page 36)
- Fields Required to Process a Sale with Cybersource Decryption (on page 37)
- Simple Order Example: Process a Sale with Cybersource Decryption (on page 38)

A sale bundles an authorization and capture in a single transaction. Request the authorization and capture at the same time. The authorization and capture amounts must be the same.

Important: In the example, the sale transaction is made using a **Visa card**, and the processor is **Visa Platform Connect**. If you want to replicate this example for a different card or a different processor, you will need to change some of the API fields in the request message.

For more information, see Cards Supported for Apple Pay on Visa Platform Connect (on page 10).

For general information about sale transactions, see the "Standard Payments Processing" section of the *Payments Developer Guide*.

Basic Steps: Processing a Sale with Cybersource Decryption

Follow these steps to process an Apple Pay sale transaction with Cybersource decryption:

1. Create the request message with the required Simple Order API fields.
    - Use the API fields listed in Fields Required to Process a Sale with Cybersource Decryption (on page 37).
    - Refer to the example in Simple Order Example: Process a Sale with Cybersource Decryption (on page 38).
2. Set the **ccAuthService_run** field to `true`.
3. Send the request to `https://ics2ws.ic3.com/commerce/1.x/transactionProcessor`.
4. Verify the response messages to make sure that the request was successful. A value of `ACCEPT` for the decision field indicates success.

Fields Required to Process a Sale with Cybersource Decryption

As a best practice, include these Simple Order fields in your request for a combined authorization and capture (sale) transaction with the Cybersource decryption implementation of Apple Pay.

Depending on your processor, your geographic location, and whether the relaxed address verification system (RAVS) is enabled for your account, some of these fields might not be required. It is your responsibility to determine whether an API field can be omitted from the transaction you are requesting. For information about the relaxed requirements for address data and expiration dates in payment transactions, see the *Payments Developer Guide*.

billTo_city

billTo_country

billTo_email

billTo_firstName

billTo_lastName

billTo_postalCode

billTo_state

billTo_street1

encryptedPayment_data

Set the value to the encrypted payment data value returned by the Full Wallet request.

encryptedPayment_descriptor

Set the value to `RklEPUNPTU1PTi5BUFBMRS5JTkFQUC5QQVlNRU5U`, the format of the encrypted payment data for Apple Pay.

encryptedPayment_encoding

Set the value to `Base64`, the encoding method for Apple Pay transactions.

paymentSolution

Set the value to `001` to identify Apple Pay as the digital payment solution.

purchaseTotals_currency

purchaseTotals_grandTotalAmount

Simple Order Example: Process a Sale with Cybersource Decryption

Request

```
ccAuthService_run=true
merchantReferenceCode=TC_1231223
paymentSolution=001
purchaseTotals_grandTotalAmount=10
purchaseTotals_currency=GBP
billTo_firstName=John
billTo_lastName=Doe
billTo_street1=901 Metro Center Blvd
billTo_city=Foster City
billTo_state=CA
billTo_postalCode=94404
billTo_country=US
billTo_email=test@cybersource.com
encryptedPayment_data=eyJkYXRhIjoiSnN0R3lkVWsyRG1mWm9YMUxNWUVoVXNicFJvTTRJMGVGK3VnbkFjUmEwczNnS0xicl
c0MVFWb1BhWEtWXC9BSU0wNWlRNitEcm1UVXUyTnVQRE91UkxIeHpkOHhqUXpkeHhQYXdMMURcL1FEa2pmakJvY2d6
OEc5UG1KanpiY1ZlZm5EWmd6UjlaNExlT3NJVlAzaG9HcEVMYWVnV3pTRXhBNEtIcDlmODdESm11WXRGVExIbW5qSH
Z0UytFa1FuWU00OGJLbnN4NGR1dm5RUVordFVwak9vOFEyZEplQ2s5WWFsSmY3NXVQaEtIVkRQVmg2dE12WExuWmpn
YjYyZXlzNytOdGlXK1dRWHJydnJ0TmxVVFwvSDYyR2x6VGxFeHBTK2FGdWVqZ3k3d2FjK3B0UERWWHR1XC9IbGdPck
5aTThcL0o4SFhCZ04yOGd5MEJuVzYxcnc1WlJ1ZWxFODNWc1FcLzlMUmVHczF0anA2aE0wZ01cLzJhaTFod2pkQzcy
VmtWNmxTT0FzZ0s5SVB0bHpPTm4rekd2aEwwTER3K01temV3VFwvNWczT3FGeHFtbURHODFqTTJ1TFlcLzd2WURlXC
9nY2lPXC9najFwa2xoTmJsXC9RPT0iLCJ2ZXJzaW9uIjoiRUNfdjEiLCJoZWFkZXIiOnsiYXBwbGljYXRpb25EYXRh
IjoiNDE3MDcwNkM2OTYzNjE3NDY5NkY2RTQ0NjE3NDYxIiwidHJhbnNhY3Rpb25JZCI6IjU0NzI2MTZFNzM2MTYzNz
Q2OTZGNkU0OTQ0IiwiZXBoZW1lcmFsUHVibGljS2V5IjoiTUlJQlN6Q0NBUU1HQnlxR1NNNDlBZ0V3Z2ZjQ0FRRXdM
QVlIS29aSXpqMEJBUUloQVBcL1wvXC9cLzhBQUFBQkFBQUFBQUFBQUFBQUFcL1wvXC9cL1wvXC9cL1
wvXC9cL01Gc0VJUFwvXC9cL1wvOEFBQUFCQUFBQUFBQUFBQUFBQUFBQVwvXC9cL1wvXC9cL1wv
XC9cL1wvXC9cLzhCQ0JheGpYWXFqcVQ1N1BydlZWMm1JYThaUjBHc014VHNQWTd6ancrSjlKZ1N3TVZBTVNkTmdpRz
V3U1RhbVo0NFJPZEpyZUJuMzZRQkVFRWF4ZlI4dUVzUWtmNHZPYmxZNlJBOG5jRGZZRXQ2ek9nOUtFNVJkaVl3cFpQ
NDBMaVwvaHBcL200N242MHA4RDU0V0s4NHpWMnN4WHM3THRrQm9ONzlSOVFJaEFQXC9cL1wvXC84QUFBQUFcL1wvXC
```

```
9cL1wvXC9cL1wvXC9cLys4NXZxdHB4ZWVoUE81eXNMOFl5VlJBZ0VCQTBJQUJENm5QV05Ta1dROVpSa2VP
b3NJV2c1
RFF6S3JoQ0ZIN1NsOHB6Z3VKZFR0OHU0N0o5aW42UnZTVGJGTHdXWXQrd2pFRkh2ZWFtaHlKeDlZSXNKTU
lJQT0iLC
JwdWJsaWNLZXlIYXNoIjoiVkFXMVlEQXRFS3dVcWpZbEtOVDNZeFlHYysxSXZ3M1RrWm9kNW9NSGd2TT0i
fSwic2ln
bmF0dXJlIjoiTUlJRFFnWUpLb1pJaHZjTkFRY0NvSUlETXpDQ0F5OENBUUV4Q3pBSkJnVXJEZ01DR2dVQU
1Bc0dDU3
FHU0liM0RRRUhBYUNDQWlzd2dnSW5NSUlCbEtBREFnRUNBaEJjbCtQZjMrVTRwazEzblZEOW53UVFNQWtH
QlNzT0F3
SWRCUUF3SnpFbE1DTUdBMVVFQXg0Y0FHTUFhQUJ0QUdFQWFRQkFBSFlBYVFCekFHRUFMZ0JqQUc4QWJUQW
VGdzB4Tk
RBeE1ERXdOakF3TURCYUZ3MHlOREF4TURFd05qQXdNREJhTUNjeEpUQWpCZ05WQkFNZUhBQmpBR2dBYlFC
aEFHa0FR
QUIyQUdrQWN3QmhBQzRBWXdCdkFHMHdnWjh3RFFZSktvWklodmNOQVFFQkJRQURnWTBBTUlHSkFvR0JBTk
M4K2tndG
dtdldGMU96amdETnJqVEVCUnVvXC81TUt2bE0xNDZwQWY3R3g0MWJsRTl3NGZJWEpBRDdGZk83UUtqSVhZ
TnQzOXJM
eXk3eER3YlwvNUlrWk02MFRaMmlJMXBqNTVVYzhmZDRmek9wazNmdFphUUdYTkxZcHRHMWQ5VjdJUzgyT3
VwOU1Nbz
FCUFZyWFRQSE5jc005OUVQVW5QcWRiZUdjODdtMHJBZ01CQUFHalhEQmFNRmdHQTFVZEFRUlJNRStBRUha
V1ByV3RK
ZDdZWjQzMWhDZzdZRlNoS1RBbk1TVXdJd1lEVlFRREhod0FZd0JvQUcwQVlRQnBBRUFBZGdCcEFITUFZUU
F1QUdNQW
J3QnRnaEJjbCtQZjMrVTRwazEzblZEOW53UVFNQWtHQlNzT0F3SWRCUUFEZ1lFQWJVS1lDa3VJS1M5UVEy
bUZjTVlS
RUltMmwrWGc4XC9KWHYrR0JWUUprT0tvc2NZNGlOREZBXC9iUWxvZ2Y5TExVODRUSHdOUm5zdlYzUHJ2N1
JUWTgxZ3
EwZHRDOHpZY0FhQWtDSElJM3lxTW5KNEFPdTZFT1c5a0prMjMyZ1NFN1dsQ3RIYmZMU0tmdVNnUVg4S1hR
WXVaTGsy
UnI2M044QXBYc1h3QkwzY0oweGdlQXdnZDBDQVFFd096QW5NU1V3SXdZRFZRUURIaHdBWXdCb0FHMEFZUU
JwQUVBQW
RnQnBBSE1BWVFBdUFHTUFid0J0QWhCY2wrUGYzK1U0cGsxM25WRDlud1FRTUFrR0JTc09Bd0lhQlFBd0RR
WUpLb1pJ
aHZjTkFRUJCUUFFZ1lBVVY2amlDOWVFOHdaQUhRRjJoZzB1aXBJdm9BWWdycmN5YlwvVUc3R2dhbWlicz
JQcEtpSW
JnSFZ4THJMNFBxU0FtSllHMHFMcFA0dU84Nloxb1VpNUdnK2xaNHZKM0NZRUt5bVpjanFGT3QwZkg0a2lP
SStlRFRO
SFhac05uV0lCZnkzV3Uwdlk3UUpXMzN6M1Y3alZWdzlqYlFEdllzbXBIelAxWmp2ZlhhUT09In0=
encryptedPayment_descriptor=RklEPUNPTU1PTi5BUFBMRS5JTkFQUC5QQVlNRU5U
encryptedPayment_encoding=Base64
```

Response to a Successful Request

```
{
}
```

Process an Apple Pay Sale with Merchant Decryption

The topics in this section shows you how to process an Apple Pay sale with the *merchant decryption* implementation of Apple Pay.

- Basic Steps: Processing a Sale with Merchant Decryption (on page 40)
- Fields Required to Process a Sale with Merchant Decryption (on page 41)
- Simple Order Example: Process a Sale with Merchant Decryption (on page 42)

A sale bundles an authorization and capture in a single transaction. Request the authorization and capture at the same time. The authorization and capture amounts must be the same.

Important: In the example, the sale transaction is made using a **Visa card**, and the processor is **Visa Platform Connect**. If you want to replicate this example for a different card or a different processor, you will need to change some of the API fields in the request message.

For more information, see Cards Supported for Apple Pay on Visa Platform Connect (on page 10).

For general information about sale transactions, see the "Standard Payments Processing" section of the *Payments Developer Guide*.

Basic Steps: Processing a Sale with Merchant Decryption

Follow these steps to process an Apple Pay sale transaction with merchant decryption:

1. Create the request message with the required Simple Order API fields.
    - Use the API fields listed in Fields Required to Process a Sale with Merchant Decryption (on page 41).
    - Refer to the example in Simple Order Example: Process a Sale with Merchant Decryption (on page 42).
2. Set the **ccAuthService_run** field to `true`.
3. Send the request to `https://ics2ws.ic3.com/commerce/1.x/transactionProcessor`.
4. Verify the response messages to make sure that the request was successful. A value of `ACCEPT` for the decision field indicates success.

Fields Required to Process a Sale with Merchant Decryption

These Simple Order API fields are required to process an Apple Pay authorization and capture as a single transaction with the merchant decryption implementation of Apple Pay.

As a best practice, include these Simple Order fields in your request for a combined authorization and capture (sale) transaction with the merchant decryption implementation of Apple Pay.

Depending on your processor, your geographic location, and whether the relaxed address verification system (RAVS) is enabled for your account, some of these fields might not be required. It is your responsibility to determine whether an API field can be omitted from the transaction you are requesting. For information about the relaxed requirements for address data and expiration dates in payment transactions, see the *Payments Developer Guide*.

billTo_city

billTo_country

billTo_email

billTo_firstName

billTo_lastName

billTo_postalCode

billTo_state

billTo_street1

card_accountNumber

Set the value to the customer's payment network token value.

ccAuthService_commerceIndicator

Set the value to the indicator that corresponds to the ecommerce value extracted from the Apple Pay response payload:

- `aesk`: American Express
- `dipg`: Discover
- `js:` JCB
- `spa`: Mastercard
- `vbv`: Visa

ccAuthService_networkTokenCryptogram

Set this field to the network token cryptogram.

paymentNetworkToken_transactionType

Set the value to `1` to identify the payload as a network token delivered from a customer's wallet.

paymentSolution

Set the value to `001` to identify Apple Pay as the digital payment solution.

purchaseTotals_currency

purchaseTotals_grandTotalAmount

token_expirationMonth

token_expirationYear

Simple Order Example: Process a Sale with Merchant Decryption

Request

```
ccAuthService_run=true
merchantReferenceCode=TC_1231223
paymentSolution=001
ccAuthService_commerceIndicator=vbv
card_accountNumber=4111111111111111
token_expirationMonth=12
token_expirationYear=2031
ccAuthService_networkTokenCryptogram=AQAAAAAACyWowDMAAAAAgQVgAgA=
paymentNetworkToken_transactionType=1
purchaseTotals_grandTotalAmount=10
purchaseTotals_currency=USD
billTo_firstName=John
billTo_lastName=Doe
billTo_street1=901 Metro Center Blvd
billTo_city=Foster City
billTo_state=CA
billTo_postalCode=94404
billTo_country=US
billTo_email=test@cybs.com
```

Response to a Successful Request

```
{
}
```

Reverse an Apple Pay Payment Authorization

The topics in this section shows you how to reverse an Apple Pay payment authorization.

- Basic Steps: Reversing a Payment Authorization (on page 43)
- Fields Required to Reverse a Payment Authorization (on page 44)
- Simple Order Example: Reverse a Payment Authorization (on page 44)

Important: In the example, the authorization being reversed is for a payment made using a **Visa card**, and the processor is **Visa Platform Connect**. If you want to replicate this example for a different card or a different processor, you will need to change some of the API fields in the request message.

For more information, see Cards Supported for Apple Pay on Visa Platform Connect (on page 10).

For general information about authorization reversals, see the "Standard Payments Processing" section of the *Payments Developer Guide*.

Basic Steps: Reversing a Payment Authorization

Follow these steps to reverse an Apple Pay payment authorization:

1. Create the request message with the required Simple Order API fields.
    - Use the API fields listed in Fields Required to Reverse a Payment Authorization (on page 44).
    - Refer to the example in Simple Order Example: Reverse a Payment Authorization (on page 44).
2. Set the **ccAuthReversalService_run** field to `true`.
3. Send the request to `https://ics2ws.ic3.com/commerce/1.x/transactionProcessor`.
4. Verify the response messages to make sure that the request was successful. A value of `ACCEPT` for the decision field indicates success.

Fields Required to Reverse a Payment Authorization

As a best practice, include these Simple Order fields in your request to reverse the authorization of an Apple Pay transaction.

Depending on your processor, your geographic location, and whether the relaxed address verification system (RAVS) is enabled for your account, some of these fields might not be required. It is your responsibility to determine whether an API field can be omitted from the transaction you are requesting. For information about the relaxed requirements for address data and expiration dates in payment transactions, see the *Payments Developer Guide*.

ccAuthReversalService_authRequestID

Set this field to the request ID that was included in the response to the original authorization request.

ccAuthReversalService_run

Set this field to `true`.

merchantReferenceCode

Set this field to the value returned in the response to the original authorization.

merchantTransactionIdentifier

Identifier that you assign to the transaction.

processingInformation.paymentSolution

Set the value to `001` to identify Apple Pay as the digital payment solution.

purchaseTotals_currency

purchaseTotals_grandTotalAmount

Simple Order Example: Reverse a Payment Authorization

Request

```
ccAuthReversalService_authRequestID=652203383441016772169
ccAuthReversalService_run=true
merchantReferenceCode=482046C3A7E94F5BD1FE3C66C
merchantTransactionIdentifier=Napa Valley Vacations
purchaseTotals_currency=USD
purchaseTotals_grandTotalAmount=10.00
```

Response to a Successful Request

```
requestID=100=1094820975023470
decision=ACCEPT
reasonCode=100
ccAuthReversalReply_amount=10.00
purchaseTotals_currency=USD
ccAuthReversalReply_reasonCode=100=1094820975023470
```

Capture an Apple Pay Authorization

The topics in this section shows you how to capture an authorized Apple Pay payment transaction.

- Basic Steps: Capturing an Authorization with Cybersource Decryption (on page 46)
- Fields Required to Capture a Payment Authorization (on page 47)
- Simple Order Example: Capture a Payment Authorization (on page 47)

Important: In the example, the payment being captured was made using a **Visa card**, and the transaction is processed with the **Visa Platform Connect**. If you want to replicate this example for a different card or a different processor, you will need to change some of the API fields in the request message.

For more information, see Cards Supported for Apple Pay on Visa Platform Connect (on page 10).

For general information about capture transactions, see the "Standard Payments Processing" section of the *Payments Developer Guide*.

Basic Steps: Capturing an Authorization with Cybersource Decryption

Follow these steps to capture an Apple Pay authorization:

1. Create the request message with the required Simple Order API fields.
    - Use the API fields listed in Fields Required to Capture a Payment Authorization (on page 47).
    - Refer to the example in Simple Order Example: Capture a Payment Authorization (on page 47).
2. Set the **ccCaptureService_run** field to `true`.
3. Send the request to `https://ics2ws.ic3.com/commerce/1.x/transactionProcessor`.
4. Verify the response messages to make sure that the request was successful. A value of `ACCEPT` for the decision field indicates success.

Fields Required to Capture a Payment Authorization

As a best practice, include these Simple Order fields in your request to capture an authorized Apple Pay payment transaction.

Depending on your processor, your geographic location, and whether the relaxed address verification system (RAVS) is enabled for your account, some of these fields might not be required. It is your responsibility to determine whether an API field can be omitted from the transaction you are requesting. For information about the relaxed requirements for address data and expiration dates in payment transactions, see the *Payments Developer Guide*.

ccCaptureService_authRequestID

Set this field to the request ID of the authorization for which you are requesting this service.

ccCaptureService_run

Set this field to `true`.

merchantId

Identifier that you assign to the transaction.

merchantReferenceCode

Set the value to `merchant_ref_number` value used in corresponding authorization request.

processingInformation.paymentSolution

Set the value to `001` to identify Apple Pay as the digital payment solution.

purchaseTotals_currency

purchaseTotals_grandTotalAmount

Simple Order Example: Capture a Payment Authorization

Request

```
ccCaptureService_authRequestID=6629978499572480812782
merchantID=Napa Valley Vacations
merchantReferenceCode=TC_1231223
purchaseTotals_currency=USD
purchaseTotals_grandTotalAmount=10.00
```

Response to a Successful Request

```
ccCaptureReply_reasonCode=100
ccCaptureReply_requestDateTime=2024-11-22T173947Z
decision=ACCEPT
merchantReferenceCode=TC_1231223
purchaseTotals_currency=USD
purchaseTotals_grandTotalAmount=10.00
```

Searching for Transactions

Use the Transaction page in the Business Center to search transactions that have been (successfully or unsuccessfully) processed on your account or for one or more of your merchants. The page consists of a Search pane and a Search Results pane.

This section describes how to search *Apple Pay transactions* using the Search pane in the Transactions page.

Important: For complete information about searching *all transactions*, see the Search for Transactions topics in the Business Center using the Cybersource Transaction Management module You must have login credentials to view the online help.

Related Information

Getting Started with Business Center Reports

Searching for Apple Pay Transactions

Use the Transactions page in the Business Center to search for Apple Pay transactions.

Follow these steps to search for Apple Pay transactions:

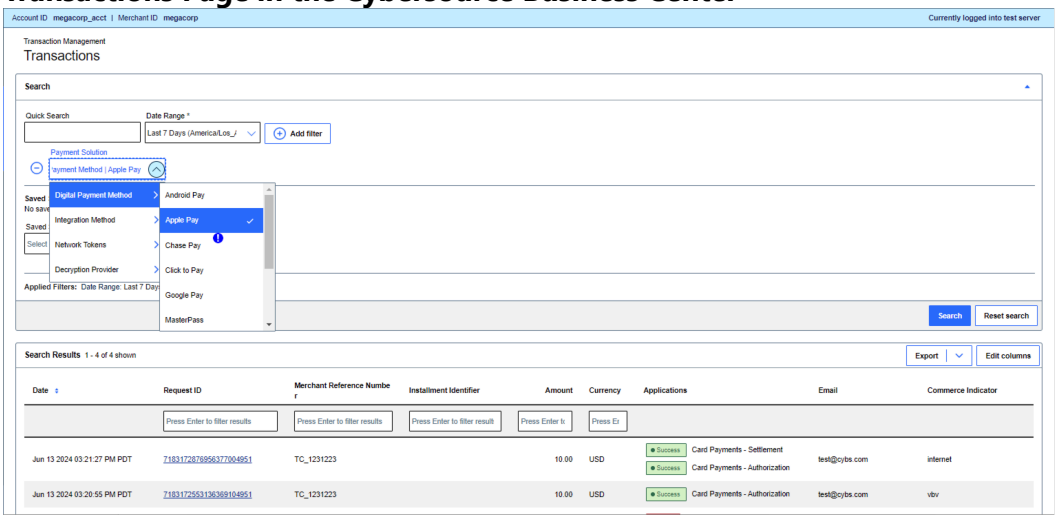
1. In the left navigation panel, click the **Transaction Management** icon and **Transactions**.

2. In the Transactions page, click Add Filter.

    a. In the New Filter drop-down box, select **Payment Solution**.

    b. In the Payment Solution drop-down box, select **Digital Payment Method** and then select **Apple Pay**.
    The Transactions page appears.

Transactions Page in the Cybersource Business Center



3. Click **Search**.

    a. Enter values into the search filters you want to apply to the list.

    b. To specify additional filters, click **Add filter**, select a filter in the list, and enter a value.

4. To download an invoice to a file, click the download icon for that invoice.

5. Click **Search**.

Reference Information

This section contains helpful information for integrating Apple Pay and for processing Apple Pay authorization transactions.

- Quick Integration for the Cybersource Decryption Method (on page 52)
- Endpoints for Services Supported with Apple Pay (on page 54)
- Optional Features Supported with Apple Pay (on page 55)

Quick Integration for the Cybersource Decryption Method

If you are integrating the **Cybersource decryption method** of Apple Pay and you are experienced in creating Apple Pay payment processing certificates, you can use these abbreviated instructions instead of the more detailed steps in Integrating Apple Pay into Your System (on page 17).

Follow these steps to integrate the Cybersource decryption method of Apple Pay:

1. Enroll your organization in the Apple Developer Program and register a merchant ID for each environment, such as sandbox and production.

2. Log in to your Cybersource merchant account at the Business Center.
   Production: `https://businesscenter.cybersource.com`
   Production in India: `https://businesscenter.in.cybersource.in.com`
   Test: `https://businesscentertest.cybersource.com/ebc2`

3. In the left navigation panel, select **Payment Configuration** and click **Digital Payment Solution**.
   The Digital Payments page appears. If you do not have the correct permissions enabled on your account, the **Digital Payment Solution** option does not appear on the left navigation panel.

4. Click **Configure**. The Apple Pay Registration page appears.

5. Enter your Apple merchant ID, and then click **Generate New Certificate Signing Request**.

6. To download the newly generated certificate signing request (CSR), click the Download icon next to the key and follow your browser's instructions to save and open the file.

7. Log into your Apple Developer account and use your CSR to create an Apple payment processing certificate that is associated with your merchant ID.

    - For Apple Pay in an iOS app, refer to *Setting Up Apple Pay (Apple Pay and Wallet)*:

      https://developer.apple.com/documentation/passkit/setting-up-apple-pay

    - For Apple Pay on the web, refer to *Configuring Your Environment (Apple Pay on the Web)*:

      https://developer.apple.com/documentation/apple_pay_on_the_web/configuring_your_environment

8. If you are validating your integration in your test environment, perform these tasks:

    a. Add test cards to your Apple sandbox tester account.
    For details, see Requirements for End-to-End Testing (on page 16).

    b. Submit Apple Pay test transactions.
    For examples of Apple Pay transactions, see Processing Apple Pay Transactions (on page 27).

    c. Adjust your integration settings as needed until your test transactions complete successfully.

9. If you are ready to integrate Apple Pay into your production environment, repeat Steps 2 through 7 with your Cybersource and Apple production accounts.

Endpoints for Services Supported with Apple Pay

This topic lists the Simple Order API endpoints that you use to request the services supported with Apple Pay.

Authorizing a Payment

Set the **ccAuthService_run** field to `true`.

Send the request to `https://ics2ws.ic3.com/commerce/1.x/transactionProcessor`.

Processing a Sale

Set the **ccAuthService_run** field to `true`.

Set the **ccCaptureService_run** field to `true`.

Send the request to `https://ics2ws.ic3.com/commerce/1.x/transactionProcessor`.

Reversing an Authorization

Capturing a Payment

Set the **ccCaptureService_run** field to `true`.

Send the request to `https://ics2ws.ic3.com/commerce/1.x/transactionProcessor`.

Processing a Follow-On Credit

Set the **ccCreditService_run** field to `true`.

Send the request to `https://ics2ws.ic3.com/commerce/1.x/transactionProcessor`.

Processing a Void

Set the **voidService_run** field to `true`.

Send the request to `https://ics2ws.ic3.com/commerce/1.x/transactionProcessor`.

Optional Features Supported with Apple Pay

These optional features are supported with Apple Pay on Visa Platform Connect.

Merchant-Initiated Transactions

Merchants can initiate a payment on the behalf of a customer. This type of transaction is called a merchant-initiated transaction (MIT). When initiating a MIT, the customer is not present. Customers must authorize the storage of their credentials and the use of these credentials for future payments.

Merchant-initiated transactions are covered in the "Standard Payments Processing" section of the *Payments Developer Guide*.

Multiple Partial Captures

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

Multiple partial captures are covered in the "Standard Payments Processing" section of the *Payments Developer Guide*.

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. Recurring payments are also known as subscriptions.

Recurring payments are covered in the "Standard Payments Processing" section of the *Payments Developer Guide*.

Subsequent Authorizations

This feature enables you to manage recurring transactions. After a successful initial authorization, you can request subsequent authorizations and request one capture for both authorizations.

Subsequent authorizations are covered in the "Standard Payments Processing" section of the *Payments Developer Guide*.

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.