**CyberSource Storefront Reference Architecture LINK Cartridge Developer Guide**

***Version*** *18.1*



­

Table of Contents

[1. Introduction to StoreFront Reference Architecture 3](#_Toc528666434)

[2. CyberSource Cartridge Overview 3](#_Toc528666435)

[3. CyberSource SFRA Cartridge Architecture 4](#_Toc528666436)

[4. Installation Guide 5](#_Toc528666437)

[4.1. Workspace Preparation 5](#_Toc528666438)

[5. Cartridge Installation 6](#_Toc528666439)

[6. Tax Calculation 8](#_Toc528666440)

[7. Credit Card Authorization 10](#_Toc528666441)

[8. Delivery Address Verification 12](#_Toc528666442)

[9. Address Verification Service (AVS) 13](#_Toc528666443)

[10. Device Fingerprint 14](#_Toc528666444)

[11. Decision Manager 15](#_Toc528666445)

[12. Decision Manager Order Update Job 16](#_Toc528666446)

[13. Payment Tokenization 17](#_Toc528666447)

[14. Subscription Token Creation 18](#_Toc528666448)

[15. Apple Pay 19](#_Toc528666449)

[16. Payer Authentication 21](#_Toc528666450)

[17. Secure Acceptance Hosted Checkout – iFrame 23](#_Toc528666451)

[18. Secure Acceptance Checkout API 26](#_Toc528666452)

[19. Secure Acceptance Flex MicroForm 28](#_Toc528666453)

[20. Credit Card Capture Service 29](#_Toc528666454)

[21. Customization 31](#_Toc528666455)

# Introduction to StoreFront Reference Architecture

The basic tenants of Salesforce Commerce Cloud’s StoreFront Reference Architecture focus on the idea of code separation and modularization. SRFA allows vendors to develop cartridges that extend, append, or replace code in the base storefront cartridge. Applying these ideas individually to views, models, and controllers, we are able to customize any part of the storefront without directly modifying the code in the storefront cartridge. This grants clients the option to pull bug fixes, and feature updates that SFCC periodically provides to the storefront, without the need to re-integrate 3rd party cartridges. It also allows 3rd party cartridge vendors to deliver updates to their cartridges, that can be quickly and easily integrated by existing customers. As a rule, a cartridge integration cannot require any changes to the base storefront cartridge. As such, you should find the integration of this, and other SFRA cartridges to be less time consuming than you may have experienced on legacy platforms.

# CyberSource Cartridge Overview

The CyberSource package contains four cartridges. A core cartridge (**int\_cybersource**) that contains core API integrations, including the building and handling of API requests, and parsing responses into objects usable by the storefront. The two legacy architecture cartridges (**int\_cybersource\_pipelines, int\_cybersource\_controllers**) each contain sets of wrappers that connect the core code to their respective SFCC platforms. If you are integrating CyberSource with a controller or pipeline version of Site Genesis, please disregard this document, and refer to the corresponding integration guide for your version of Site Genesis.

Version 18.1 of the CyberSource cartridge package adds a fourth cartridge (**int\_cybersource\_sfra/ LINK\_cybersource**), which combines a modified version of the core code that exists in the int\_cybersource, along with the necessary hooks, extensions, and wrappers to connect this code to the SFRA storefront. When integrating CyberSource with SFRA, you should only upload the LINK\_cybersource cartridge to your workspace and storefront. The remaining three cartridges are not utilized in this integration and can be ignored. The following pages describe the high-level architecture of the SFRA CS Architecture, along with details regarding specific Integrations.

# CyberSource SFRA Cartridge Architecture

****

This high-level diagram displays the key connection points and data paths through the various services provided by CyberSource. The LINK cartridge connects many services to the storefront via hooks made available by the SFRA storefront. Some Services, such as AVS, DAV, and Fraud Detection do not utilize their own API requests, and are handled in the Authorization, or Payment Processor API Requests. An overview of each service, along with usage details, is provided below.

# Installation Guide

## Workspace Preparation

Out of the box, this cartridge (int\_cybersource\_sfra) assumes it has been placed in the same directory as the storefront-reference-architecture folder. If you have a different project set-up, you will need to open the file ‘int\_cybersource\_sfra/package.json’ and modify the paths.base value to point to your ‘app\_storefront\_base’ cartridge. This path is used by the JS and SCSS build scripts. Once complete, follow the below steps.

1. On your terminal, navigate to int\_cybersource\_sfra
2. If you have not already, install node using ‘nvm install node’
3. Run ‘npm install’ to install all of the local dependencies.
4. If using Eclipse, add the ‘Link\_cybersource cartridge to your workspace.
5. If using Visual Studio Code, do the following:
   1. Create the file `dw.json` under the int\_cybersource\_sfra folder with contents:

{

"hostname": "your-sandbox-hostname.demandware.net",

"username": "yourlogin",

"password": "yourpwd",

"version": "version\_to\_upload\_to",

"cartridge": [

"LINK\_cybersource",

"app\_storefront\_base",

"modules"

]

}

Note: Configuration of this file, and the cartridges you want uploaded and watched will depend on your project.

* 1. Run `npm run uploadCartridge` command to upload cartridges defined in dw.json to the server defined in dw.json.

1. Run `npm run compile:scss` to compiles all .scss files into CSS.
2. Run `npm run compile:js` - Compiles all .js files and aggregates them.
3. If using Eclipse, refresh your project contents, as new JS and css files may have been created, that need to be uploaded. To avoid doing this every time you compile your JS or SCSS, Enable the Workspace Preference ".

Note: If you have trouble getting your build scripts to run, or encounter errors in your npm install, try setting your node version to 8.11.3 and go back to step 3.

# Cartridge Installation

Whether installing the cartridge for the first time, or upgrading to a new version, perform the following steps to ensure you have the latest metadata.

After following the above steps, the LINK\_cybersource cartridge should be uploaded to your SFCC instance, to the active code version. Follow the steps below to configure your server for the LINK\_cybersource cartridge.

1. In Business Manager, navigate to ‘Administration > Sites > Manage Sites > Your Site’ and select the ‘Settings’ tab.
2. Add ‘LINK\_cybersource:’ to the left side of the cartridge path, before ‘app\_storefront\_base’
3. Navigate to ‘Administration > Site Development > Import & Export’.
4. Upload and import the meta data file:
   * int\_cybersource\_sfra/configuration/CS SFRA Metadata v18\_1.xml
5. Navigate to ‘Administration > Operations > Import & Export’
6. Upload and import the services data file:
   * int\_cybersource\_sfra/configuration/CS SFRA Services v18\_1.xml
7. Navigate to ‘Merchant Tools > Site Preferences > CyberSource: Core’
8. Enter the Merchant ID that CyberSource provided you with in the ‘CyberSource Merchant ID’ field.
9. Generate a SOAP API key in the CyberSource Business Center.
   * Log into CyberSource Business Center.
   * Navigate to ‘Account Management > Transaction Security Keys’
   * Select ‘Security Keys for the SOAP Toolkit API’
   * Click the ‘Generate Key’ button.
   * Copy the key at the bottom of the page and save it somewhere secure. You may need it later.
10. Navigate back to the CyberSource: Core Custom Site Preferences.
11. Enter the Merchant Key that you just created in the ‘CyberSource Merchant Key’ field.
12. Select the appropriate ‘CyberSource Endpoint’ for your instance. Production should point to the ‘Production’ CS endpoint, and all other instances should point to the ‘test’ CS endpoint.
13. Enter your Developer ID, and Merchant ID supplied by CyberSource, in the corresponding fields.
14. Scroll to the top of the page and click ‘Save’.
15. Basic cartridge installation is complete. Reference the ‘Usage’ and ‘Configuration’ guides on the following pages for activation and configuration of individual services. Some services will require importing additional data files to your sandbox. Instructions to do so are included those specific services ‘Usage’ sections.

# Tax Calculation

*Integration Overview*

The CyberSource tax service is integrated via the SFRA OOTB dw.order.calculateTax hook. The calculateTax hook is registered in the hooks.json file with script ./cartridge/scripts/hooks/tax/taxes. This script acts as a wrapper to the core CyberSource Tax code. Tax values are retrieved from CyberSource and updated in the basket. The hook returns a Status object, preventing the OOTB calculateTax hook from being called, thus taking tax calculation priority away from the storefront cartridge. If you intend to use CyberSource as your tax calculator, you should not have another cartridge ahead of this one, that also calls the calculateTax hook. In the case of an error, or unresponsive endpoint, the OOTB calculateTax script will be used, as a back-up.

Product information is provided on an individual line item basis and all merchant/request IDs are captured for future reference. When the customer enters shipping information, the Tax Service is called to calculate taxes. Taxes will only be recalculated when a change has been made to the cart that can affect the total tax amount.

*Implementation*

To use CyberSource Tax services on SFRA, ensure you have followed all steps in the "Cartridge Installation" guide above. A CyberSource Merchant ID, and CyberSource Merchant Key are required for this service. Enter these values in the corresponding site preferences under the "CyberSource" group. Work with CyberSource to ensure tax services are activated and functioning on your account. Follow the steps below to configure the service in Business Manager.

1. Merchandise the taxCode field on all products in your catalog.
2. Optiponally, you can set the ‘CS Tax Calculation Default Product Tax Code’ site preference to act as a fallback tax code for all products without a merchandised tax code.
3. Determine and set all associated site preference listed below.
4. Set the ‘CS Tax Calculation Enabled’ site preference to ‘Yes’ to enable the service.

*Configuration*

Site Preference Group: CyberSource: Core

|  |  |
| --- | --- |
| **Preference Name** | **Usage** |
| CS Tax Calculation Enabled | Enable or disable CyberSource tax service. |
| CS Tax Calculation Nexus States List | List of states to charge tax in. |
| CS Tax Calculation No Nexus States List | List of States to not charge tax in. |
| CS Tax Calculation Default Product Tax Code | Default tax code used when tax code is not set on a product. |
| CS Tax Calculation Purchase Order Acceptance City | Purchase order acceptance state code. |
| CS Tax Calculation Purchase Order Acceptance Zip Code | Purchase order acceptance zip code. |
| CS Tax Calculation Purchase Order Acceptance Country Code | Purchase order acceptance country code. |
| CS Tax Calculation Purchase Order Origin City | Purchase order origin city. |
| CS Tax Calculation Purchase Order Origin StateCode | Purchase order origin state code. |
| CS Tax Calculation Purchase Order Origin ZipCode | Purchase order origin zip code. |
| CS Tax Calculation Purchase Order Origin Country Code | Purchase order origin country code. |
| CS Tax Calculation ShipFrom City | Ship from city. |
| CS Tax Calculation ShipFrom StateCode | Ship from state code. |
| CS Tax Calculation ShipFrom ZipCode | Ship from zip code. |
| CS Tax Calculation ShipFrom Country Code | Ship from country code. |

# Credit Card Authorization

*Integration Overview*

The CC Auth service is integrated via the SFRA OOTB dynamically generated app.payment.processor.cybersource\_credit hook. The cybersource\_credit hook is registered in the hooks.json file with script ./cartridge/scripts/hooks/payment/processor/cybersource\_credit. This script acts as a wrapper to the core CyberSource Authorization code. Behind this wrapper, an API request is constructed, sent to CS, and the response parsed. In the case of a successful authorization (response code 100), the hook returns a JSON object without an error. All other response codes received result in an error being present in the return object, triggering the storefront to display an error message, and not create the order. Actions taken when making the Authorization call are as follows:

1. Creates CyberSource authorization request using ship-to, bill-to, credit card data, and purchase total data from the current basket.
2. If authorize Payer is configured, then make the authorize payer request. If not, ignore and continue with the authorization request.
3. Create a credit card authorization request.
4. If DAV is enabled, set up DAV business rules, as needed.
5. Set up AVS if enabled.
6. Make the service call to CyberSource via the SOAP API.
7. If Delivery Address Verification is enabled, then:
   1. Capture pertinent DAV result information & DAV Reason Code. Update shipping address if a suggestion was returned and the ‘CS DAV Update Shipping Address With DAV Suggestion’ site preference is enabled.
   2. If DAV fails and DAV On Failure is set to ‘REJECT’, then exit immediately with rejection response
8. If DAV On Failure is set to ‘APPROVE’ and the DAV Reason Code is a fail code (not 100), then:
   1. Exit immediately with declined or review response, as merchant defines
9. Capture pertinent AVS information.
10. Capture Fraud response in a session variable to be handled later.
11. Validate authorization reason code and set corresponding values, based on Auth response code.

The list of activities depicted in the following diagram takes place when API request is made for an online credit card authorization: [**Source, CyberSource Credit Card Service, and October 2009**]



1. The customer places an order and provides the credit card number, the card expiration date, and other information about the card.
2. You send a request for authorization over a secure Internet connection. If the customer buys a digitally delivered product or service, you can request both the authorization and the capture at the same time. If the customer buys a physically fulfilled product, do not request the capture until you ship the product.
3. CyberSource validates the order information, and then contacts your payment processor and requests authorization.
4. The processor sends the transaction to the card association, which routes it to the issuing bank for the customer’s credit card. Some card companies, including Discover and American Express, act as their own issuing banks.
5. The issuing bank approves or declines the request. Depending on the card type, the bank could also use the Address Verification Service (AVS) to determine whether the customer provided the correct billing address. For more information about AVS, refer to AVS service documents via the CyberSource Services Documentation at <http://www.cybersource.com/support_center/support_documentation/services_documentation/payment.php> or as described in this integration guide.
6. CyberSource runs its own tests, and then tells you if the authorization succeeded.
7. Response is sent back to the client.

*Implementation*

To use CyberSource Tax services on SFRA, ensure you have followed all steps in the "Cartridge Installation" guide above. A CyberSource Merchant ID, and CyberSource Merchant Key are required for this service. Enter these values in the corresponding site preferences under the "CyberSource: Core" group and work with CyberSource to ensure tax services are activated and functioning on your account. Follow the steps below to configure the service in Business Manager.

1. Import ‘int\_cybersource\_sfra/configuration/CS SFRA PaymentMethods v18\_1.xml’ into your sandbox.
2. Under 'Merchant Tools > Ordering > Payment Methods' Make sure the 'CREDIT\_CARD' payment method is enabled and configured to use the CYBERSOURCE\_CREDIT payment processor.
3. On the same page, select 'Credit/Debit cards' to enable the credit card types you want supported.

*Configuration*

Other than the standard requirements of all CyberSource services covered in the installation guide, there are no site preferences associated with a basic Authorization call. But, many services utilize the Credit Card Authorization API to communicate with CyberSource. Please see the sections of any of these services you may be using, to determine how to configure them. i.e. DAV, AVS, Decision Manager.

# Delivery Address Verification

*Integration Overview*

The Delivery Address Verification does not make and independent API call to validate address data. The shipping address is attached to the payment processing API call and verified in the same request. This means, in its current state, the service cannot be used as a stand-alone address validator on shipping address submission. The validation does not happen until the payment processor is called, which is after the customer has confirmed their order details. At the time of the authorization, a payment processing API call to CyberSource will attempts to validate the address. If the address is valid, a ‘100’ response code is returned, and the order is processed by the storefront as usual. If CyberSource was able to correct mistakes in the supplied address, this corrected address will be sent back in the response. If the appropriate site preference is selected, this address will be automatically copied into the order shipping address, overriding the customers entry. If a corrected address could not be determined, an error will be passed back to the storefront, and the order is either created or not, based on a site preference value.

*Implementation*

To use the DAV service, ensure you have followed all steps in the "Cartridge Installation" guide above. As this service runs through the Authorization API, a CyberSource Merchant Id and CyberSource Merchant Key are required. Enter these values in the corresponding site preferences under the "CyberSource" group and work with CyberSource to ensure the DAV service is activated and functioning on your account. To enable this service, follow the below steps:

1. In the site preference group ‘CyberSource: Core’, set the ‘CS DAV Delivery Address Verification Enabled’ preference to ‘Yes’
2. Configure the site preferences listed below based on your business needs.

*Configuration*

Site Preference Group: CyberSource: Core

|  |  |
| --- | --- |
| **Preference Name** | **Usage** |
| CS DAV Delivery Address Verification Enabled | Enable or disable the DAV service. |
| CS DAV Update Shipping Address With DAV Suggestion | Update the shipping address with the CS suggestion, if found. |
| CS DAV On Failure | Accept or Reject the order if DAV fails. |

# Address Verification Service (AVS)

*Integration Overview*

AVS does not exist as a stand-alone callable service. The service is performed during an Authorization request. Please refer to the Credit Card Authorization Service overview to understand how AVS has been integrated.

*Implementation*

Assuming you have implemented the Credit Card Authorization service, you are ready to use the AVS service. Configure the below site preferences to suite your business needs.

*Configuration*

Site Preference Group: CyberSource: Core

|  |  |
| --- | --- |
| **Preference Name** | **Usage** |
| CS AVS Ignore AVS Result | Effectively enables or disables the AVS service. |
| CS AVS Decline Flags | Leave empty to follow CS default decline flag strategy.  Enter flags separated by commas to overwrite the default flag rules. |

# Device Fingerprint

*Integration Overview*

Device Fingerprint collection is handled in the htmlhead.isml template. An include in the <head> of every page calls the CYBDeviceFingerprint-GetFingerprint endpoint. This controller uses a session variable to remember the last time a device fingerprint was generated for this session. If one has not been created in a time greater than a given site preference value, a flag is sent to the rendering template to generate a new one. The session variable is then updated with the current time. In this way, every user will have a device fingerprint generated on their first visit, and the device fingerprint for the user will be updated every N milliseconds. By default, device fingerprints expire every 24 hours, and a customer session can be longer than that, requiring this periodic regeneration.

The fingerprint is generated through a JS include in the <head> element, which pulls self-executing code from a third-party service. This data is stored as a digital fingerprint, along with the storefront session ID of the user on the 3rd party services servers. CyberSource is able to look up the fingerprint using the session ID that is passed in an Authorization request. The fingerprint can then be used in CyberSource Decision Manager rules during the Authorization.

*Implementation*

To use Device Fingerprinting, ensure you have followed all steps in the "Cartridge Installation" guide above. You will need a ThreatMetix URL to retrieve the JS that runs on the site, and an Organization ID. Work with CyberSource to obtain these values. Configure the below site preferences to activate the service.

*Configuration*

Site Preference Group: CyberSource: Core

|  |  |
| --- | --- |
| **Preference Name** | **Usage** |
| CS Device Fingerprint Enabled | Enable or Disable the Device Fingerprint Service. |
| CS Device Fingerprint Organization ID | Device Fingerprint Organization ID |
| CS Device Fingerprint ThreatMetrix URL | URL pointing to JS that generates and retrieves the fingerprint. |
| CS Device Fingerprint Time To Live | Time, in milliseconds between generating a new fingerprint for any given customer session. |

# Decision Manager

*Integration Overview*

The Decision Manager integration does not make and independent API call to retrieve a fraud decision. A flag is passed through the various payment method API requests, indicating to CyberSource whether or not to run DM rules against the transaction. The results of the decision will be passed back in the API response, and saved in “session.custom.CybersourceFraudDecision” for later use. SFRA utilities a separate hook, after payment processing, that handles fraud detection. This cartridge makes use of this hook, by subscribing to "app.fraud.detection" with script './cartridge/scripts/hooks/fraudDetection'. The fraudDetection script reads the stored value in the CybersourceFraudDecision session variable and returns the expected value back to SFRA. In the case of an ACCEPT, the hook will return a status of 'success'. In the case of a 'REVIEW' decision, the hook will return 'flag'. In the case of a 'REJECT' decision, the hook will return 'fail'. The SFRA storefront cartridge handles these responses as such:

'success' will allow the order to be created, and Order confirmation status is set to 'Confirmed'.

'flag' will allow the order to be created, and Order confirmation status is set to 'Not Confirmed'.

'fail' will prevent the order from being placed and display a general error message to the customer.

*Implementation*

To use Decision Manager, ensure you have followed all steps in the "Cartridge Installation" guide above. As this service runs through the Authorization API, a CyberSource Merchant Id and CyberSource Merchant Key are required. Enter these values in the corresponding site preferences under the "CyberSource Core" group and work with CyberSource to ensure the Decision Manager is activated and functioning on your account. To enable this service, work with CyberSource to configure your fraud rules in CS Business Center. All Decision Manager configurations are done through the CS portal, not in your storefront Business Center.

Once your Fraud rules have been configured, set the below site preference to ‘Yes’ to enable the feature on your storefront.

Note: Be sure to read the next section regarding the scheduled job associated with Decision Manager.

*Configuration*

Site Preference Group: CyberSource: Core

|  |  |
| --- | --- |
| **Preference Name** | **Usage** |
| CS Decision Manager Enabled | Enable or Disable Decision Manager |

# Decision Manager Order Update Job

*Integration Overview*

This job uses a simple API to retrieve order decisions from CyberSource and update the order confirmation status in SFCC.

As described in the previous section, when Decision Manager is enabled, a certain number of orders will be flagged for review, and not fully confirmed in your SFCC storefront. When integrating with your OMS, you must decide if you want to export orders that are set to ‘Not Confirmed’. If you only send Confirmed orders to your OMS, you will need this job to update the confirmation status of the orders that have been reviewed and accepted.

*Implementation*

If you are not using Decision Manager, you do not need this job. If your OMS want’s you to send them ‘Not Confirmed’ Orders, you may or may not want this job. You will need to determine if the Order confirmation status is required, or desired in SFCC for your business needs.

To Integrate this job into your site, follow the below steps:

1. Navigate to ‘Administration > Operations > Import & Export’
2. Import the file int\_sybersource\_sfra/configuration/CS SFRA ScheduledJobs v18\_1.xml
3. Navigate to ‘Administration > Operations > Job Schedules’
4. Select the Job ‘CyberSource: Decision Manager Order Update’
5. Select the ‘Step Configurator tab.
6. Select the Sites you want the Job to run on, from the ‘Scope’ button.
7. Click on the ‘UpdateOrderStatus’ Step.
8. Scroll down to the ‘Custom Parameters’ section.
9. Fill in the job parameters. (parameter descriptions are below)
10. Click the ‘Assign’ button.
11. Navigate to the ‘Schedule and History’ tab and configure the frequency you would like the job to run.
12. Ensure the ‘Enabled’ check box is selected.

*Configuration*

Job Parameters: UpdateOrderStatus

|  |  |
| --- | --- |
| **Preference Name** | **Usage** |
| MerchantId | Your CyberSource merchant ID (The same one used to log into CS Business Center) |
| username | A CyberSource Business Center login User name |
| password | A CyberSource Business Center login Password |

Note: The Scheduled Jobs import file will import 2 jobs. If you have not integrated Secure Acceptance, you can safely delete the ‘CyberSource: SecureAcceptance Merchant Post’ Job.

# Payment Tokenization

*Integration Overview*

Tokenization is the replacement of sensitive data with a unique identifier that cannot be mathematically reversed. When this service is enabled, a token will be generated when a customer saves a credit card. This token is used when making a payment with that saved card. Typically, the token will retain the last four digits of the card as a means of accurately matching the token to the payment card owner. The remaining numbers are generated using proprietary tokenization algorithms.

Tokenization is implemented in two places:

The Billing Step

During checkout, a registered and logged in user has the option to save the credit card they are using to make the purchase. When they choose this option, and confirm the order, the route ‘CheckoutServices- PlaceOrder’ is being replaced by a new version in the CyberSource cartridge. This script contains all of the code present in the SFRA storefront version at time of development, with additions that make an API call to CS requesting a token be generated before the Authorization is made. CyberSource will respond with a token, which is saved to the payment, and sent in the subsequent Authorization call.

The Account: Payment Instruments Page

Customers also have the option of creating and deleting payment methods from their Account Page: Payment Instrument. To achieve tokenization from here, the ‘PaymentInstruments-SavePayment route was replaced. All of the SFRA code has been copied into this script, along with an addition of making an API call to request a token and saving the token to the Payment Instrument.

Deleting a Card works similarly. The ‘PaymentInstruments-DeletePayment route has been replaced and modified to send an API request to CyberSource to delete the token, before deleting the Payment Instrument in SFCC.

*Implementation*

To use Tokenization, ensure you have followed all steps in the "Cartridge Installation" guide above. Work with CyberSource to ensure the Tokenization is activate and working on your account. Configure the below site preferences to activate the service on your storefront:

*Configuration*

Site Preference Group: CyberSource: Core

|  |  |
| --- | --- |
| **Preference Name** | **Usage** |
| CS Tokenization Enabled | Enable or Disable the Tokenization Service. |

# Subscription Token Creation

*Integration Overview*

When making an authorization call to CyberSource, you have the option to request a subscription token be created. When enabled, your authorization calls will include a request to generate this token. The token will be returned in the response object and saved to the payment instruments ‘creditCardToken’ field. You can than utilize this token in other third-party integrations. For example, you can send it to your OMS, to allow them to perform Future Authorizations, Captures, Reversals, etc.

*Implementation*

To enable subscript token generation, ensure you have followed all steps in the "Cartridge Installation" guide above. Configure the below site preference to enable the service:

*Configuration*

Site Preference Group: CyberSource: Core

|  |  |
| --- | --- |
| **Preference Name** | **Usage** |
| CS Subscription Tokens Enabled | Enable or Disable the option to generate subscription tokens. |

# Apple Pay

*Integration Overview*

The CyberSource apple pay authorization service is integrated via the SFRA OOTB “dw.extensions.applepay.paymentAuthorized.authorizeOrderPayment”.  The apple pay authorizeOrderPayment hook is registered in the hooks.json file with script “/cartridge/scripts/hooks/applepay/applePayAuth”.

This script acts as a wrapper to the core CyberSource apple pay authorization code.  Apple Pay payment card data is sent to CyberSource for authorization and authorization status is retrieved from CyberSource. The hook returns a Status object, preventing the OOTB apple pay authorizeOrderPayment hook from being called, thus taking apple pay authorization priority away from the storefront cartridge.

*Implementation*

Business Manager Configuration

1. Go to: “Merchant Tools > Site Preferences > Apple pay
2. Check “Apple Pay Enabled?”
3. Fill in the “Onboarding” form:
4. Ensure “Apple Merchant ID” and “Apple Merchant Name” match settings in your apple account
5. Ensure all other fields match the your supported CyberSource settings
6. Fill in the “Storefront Injection” form:
7. Selects where apple pay buttons should be displayed on your site.
8. Fill in “Payment Integration” form:
9. Leave all form fields blank
10. Ensure “Use Basic Authorization” is unchecked

CyberSource Business Center Configuration

1. Go to: “Account Management > Digital Payment Solution
2. Click “Sign Up” next to Apple Pay
3. Follow the step by step instructions on the page

*Configuration*

The CyberSource Apple Pay integration does not make use of any custom site preferences. See the following SFCC Apple Pay preferences for configuration:

Site Preferences: Apple Pay

|  |  |
| --- | --- |
| **Preference Name** | **Usage** |
| Apple Pay Enabled? | Enable or Disable the Apple Pay Service. |
| Apple Merchant ID | Match settings in your Apple account. |
| Apple Merchant Name | Match settings in your Apple account. |
| Country Code | Match setting in your CyberSource account. |
| Merchant Capabilities | Match setting in your CyberSource account. |
| Supported Networks | Match setting in your CyberSource account. |
| Required Shipping Address Fields | Match setting in your CyberSource account. |
| Required Billing Address Fields | Match setting in your CyberSource account. |
| Inject Apple Pay Button on Mini Cart? | Display Apple Pay button in the mini cat. |
| Inject Apple Pay Button on Cart Page? | Display Apple Pay button in the cart. |
| Redirect Pages to HTTPS? | Redirect Pages to HTTPS |
| Use Commerce Cloud Apple Pay Payment API? | Not used. Leave blank. |
| Payment Provider URL | Not used. Leave blank. |
| Payment Provider Merchant ID | Not used. Leave blank. |
| API Version | Set to v1 |
| Use Basic Authorization | This must be enabled. |
| Payment Provider User | Not used. Leave blank. |
| Payment Provider Password | Not used. Leave blank. |
| Use JWS? | Not used. Leave blank. |
| JWS Private Key Alias | Not used. Leave blank. |

# Payer Authentication

*Integration Overview*

CyberSource Payer Authentication services enable you to add support to your web store for card authentication services, including Visa Verified by VisaSM, MasterCard® and Maestro® SecureCode™ (UK Domestic and international), American Express SafeKeySM, and JCB J/Secure™.

These card authentication services deter unauthorized card use and protect you from fraudulent chargeback activity referred to as liability shift.

How It Works

Payer Authentication provides the following services:

1. Check Enrollment: Determines whether the customer is enrolled in one of the card authentication programs.
2. Validate Authentication: Ensures that the authentication that you receive from the issuing bank is valid.



The Check Enrollment service determines whether the customer is enrolled in one of the

Card authentication services:

No: If the card is not enrolled, you can process the authorization immediately.

Yes: If the card is enrolled, the customer’s browser displays a window where the customer can enter the password associated with the card. This is how the customer authenticates their card with the issuing bank.

If the password matches the password stored by the bank, you need to verify that the information is valid with the Validate Authentication service. If the identity of the sender is verified, you can process the payment with the Card Authorization service.

If the password does not match the password stored by the bank, the customer may be fraudulent. You must refuse the card and can request another form of payment.

*Implementation*

To enable Payer Authentication services on SFRA, ensure you have followed all steps in the "Cartridge Installation" guide above. A CyberSource Merchant ID, CyberSource Merchant Key and Cybersource PA Merchant ID are required for this service. Enter these values in the corresponding site preferences under the "CyberSource: Core" group and work with CyberSource to ensure required Credit card types are configured on Cybsersource portal to participate in the Payer Authentication service on your account. Also, work with Cybersource to understand if the ByPass rules are set for your account. ByPass rules define the window of order transaction amounts which then decide whether the Authentication window is presented or bypassed. Sample of Bypass rules:

1) Bypass authentication if transaction amount is > $500 and

2) Bypass authentication if transaction amount is < $100

Import Payment methods – Already covered in section Credit Card Authorization, this section is included for reference

1. Import ‘int\_cybersource\_sfra/configuration/CS SFRA PaymentMethods v18\_1.xml’ into your sandbox.
2. Under 'Merchant Tools > Ordering > Payment Methods' Make sure the 'CREDIT\_CARD' payment method is enabled and configured to use the CYBERSOURCE\_CREDIT payment processor.
3. On the same page, select 'Credit/Debit cards' to select the credit card type you want supported.

*Configuration*

Under 'Merchant Tools > Ordering > Payment Methods, select ‘Credit/Debit cards’. Select each credit card as needed and ensure custom-attribute ‘Enable Payer Authentication’ checkbox is checked.

# Secure Acceptance Hosted Checkout – iFrame

*Integration Overview*

Secure Acceptance payment gateway is used to process transaction requests directly from the customers browser so that sensitive payment data does not pass through the SFCC servers.

*Secure Acceptance iFrame*: Customer will be redirected to a Secure Acceptance payment gateway within an iFrame, embedded in a new summary page added to the checkout flow.

Secure Acceptance Web/Mobile Authorization Sequence flow:

1. 'SecureAcceptanceAuthorize' function create Request and signature using signed and unsigned field names and validate the request.
2. After successful validation of the request using signature and then ajax call is made show the to complete the checkout flow
3. After successful checkout completion, Customer is returned to Demandware custom controller method.
4. Secure Acceptance response method get the response in CurrentHttpParameterMap, again signature is created using the response data and matched with the response signature, once validated response is parsed
5. Based on Decision and reason code Order will get placed or failed in Demandware.

*Implementation*

To use CyberSource Secure Acceptance iFrame service on SFRA, ensure you have followed all steps in the "Cartridge Installation" guide above. A CyberSource Merchant ID, and CyberSource Merchant Key are required for this service. Enter these values in the corresponding site preferences under the "CyberSource" group and work with CyberSource to Secure Acceptance services are activated and functioning on your account. To complete the checkout process successfully create Secure Acceptance profile on CyberSource business center console under ' Tools & Settings > Secure Acceptance > Profiles > Create New Profile '. While creating the profile check the checkbox ' Web/Mobile’ for Integration Methods in Profile Information and configure all the mandatory settings and ensure you activate the profile. Follow the steps below to configure the service in Business Manager.

1. Determine and set all associated site preference listed below.
2. Set the ‘CsSAType’ site preference to ‘SA\_IFRAME' to enable the service.

*Configuration*

Site Preference Group: CyberSource Secure Acceptance

|  |  |
| --- | --- |
| **Preference Name** | **Usage** |
| CsSAType | Secure Acceptance Type |
| SA\_Iframe\_ProfileID | Secure Acceptance Iframe Profile ID |
| SA\_Iframe\_SecretKey | Secure Acceptance Iframe secret key |
| SA\_Iframe\_AccessKey | Secure Acceptance Iframe Access Key |
| CsSAIframetFormAction | CyberSource secure acceptance Iframe form action |
| CsSAOverrideBillingAddress | CyberSource Secure Acceptance Override Billing Address |
| CsSAOverrideShippingAddress | CyberSource Secure Acceptance Override Shipping Address |
| CsCvnDeclineFlags | CyberSource Ignore CVN Result (CVN) |

Secure Acceptance Redirect

*Integration Overview*

Secure Acceptance payment gateway is used to process transaction requests directly from the customers browser, so that sensitive payment data does not pass through SFCC servers.

*Secure Acceptance Redirect*: Customer will be redirected to a Secure Acceptance payment gateway when clicking on Place Order from Review Page

Secure Acceptance Web/Mobile Authorization Sequence flow:

1. 'SecureAcceptanceAuthorize' function create Request and signature using signed and unsigned field names and validate the request.
2. After successful validation of the request using signature and then its redirected to the secure payment page to complete the checkout flow
3. After successful checkout completion, Customer is returned to Demandware custom controller method.
4. Secure Acceptance response method get the response in CurrentHttpParameterMap, again signature is created using the response data and matched with the response signature, once validated response is parsed
5. Based on Decision and reason code Order will get placed or failed in Demandware

*Implementation*

To use CyberSource Secure Acceptance Redirect service on SFRA, ensure you have followed all steps in the "Cartridge Installation" guide above. A CyberSource Merchant ID, and CyberSource Merchant Key are required for this service. Enter these values in the corresponding site preferences under the "CyberSource" group and work with CyberSource to Secure Acceptance services are activated and functioning on your account. To complete the checkout process successfully create Secure Acceptance profile on CyberSource business center console under ' Tools & Settings > Secure Acceptance > Profiles > Create New Profile '. While creating the profile check the checkbox ' Web/Mobile’ for Integration Methods in Profile Information and configure all the mandatory settings and ensure you activate the profile. Follow the steps below to configure the service in Business Manager.

1. Determine and set all associated site preference listed below.

1. Set the ‘CsSAType’ site preference to ‘SA\_REDIRECT' to enable the service.

*Configuration*

Site Preference Group: CyberSource Secure Acceptance

|  |  |
| --- | --- |
| **Preference Name** | **Usage** |
| CsSAType | Secure Acceptance Type |
| SA\_Redirect\_ProfileID | Secure Acceptance Redirect Profile ID. |
| SA\_Redirect\_SecretKey | Secure Acceptance Redirect Secret Key. |
| SA\_Redirect\_AccessKey | Secure Acceptance Redirect Access Key. |
| CsSARedirectFormAction | CyberSource secure acceptance redirect form action. |
| CsSAOverrideBillingAddress | CyberSource Secure Acceptance Override Billing Address. |
| CsSAOverrideShippingAddress | CyberSource Secure Acceptance Override Shipping Address. |
| CsCvnDeclineFlags | CyberSource Ignore CVN Result (CVN). |

# Secure Acceptance Checkout API

*Integration Overview*

Secure Acceptance Checkout API (Silent Order POST) payment gateway is used to process transaction requests directly from the customers browser, so that sensitive payment data does not pass through SFCC servers.

Secure Acceptance Silent Order Post: Credit Card form data is posted to a Secure Acceptance silent post URL where a token is generated and returned to SFCC. The user is redirected to the order review page and standard card authorization flow continues, using the token received from SA.

Secure Acceptance Silent Order Post Authorization Sequence flow:

1. 'SecureAcceptanceAuthorize' function create Request and signature using signed and unsigned field names and validate the request.
2. After successful validation of the request using signature and then ajax call is made show the to complete the checkout flow
3. After successful checkout completion, Customer is returned to Demandware custom controller method.
4. Secure Acceptance response method get the response in CurrentHttpParameterMap, again signature is created using the response data and matched with the response signature, once validated response is parsed
5. Based on Decision and reason code Order will get placed or failed in Demandware.

*Implementation*

To use CyberSource Secure Acceptance Silent Post service on SFRA, ensure you have followed all steps in the "Cartridge Installation" guide above. A CyberSource Merchant ID, and CyberSource Merchant Key are required for this service. Enter these values in the corresponding site preferences under the "CyberSource" group and work with CyberSource to Secure Acceptance services are activated and functioning on your account. To complete the checkout process successfully create Secure Acceptance profile on CyberSource business center console under ' Tools & Settings > Secure Acceptance > Profiles > Create New Profile '. While creating the profile check the checkbox ' Silent Order Post’ for Integration Methods in Profile Information and configure all the mandatory settings and ensure you activate the profile. Follow the steps below to configure the service in Business Manager.

1. Determine and set all associated site preference listed below.
2. Set the ‘CsSAType’ site preference to ‘SA\_SILENTPOST' to enable the service.

*Configuration*

Site Preference Group: CyberSource Secure Acceptance

|  |  |
| --- | --- |
| **Preference Name** | **Usage** |
| CsSAType | Secure Acceptance Type |
| SA\_Silent\_ProfileID | Secure Acceptance Silent Post Profile ID. |
| SA\_Silent\_SecretKey | Secure Acceptance Silent Post Secret Key. |
| SA\_Silent\_AccessKey | Secure Acceptance Silent Post Access Key. |
| Secure\_Acceptance\_Token\_Create\_Endpoint | Secure\_Acceptance\_Token\_Create\_Endpoint. |
| Secure\_Acceptance\_Token\_Update\_Endpoint | Secure\_Acceptance\_Token\_Update\_Endpoint. |
| CsSAOverrideBillingAddress | CyberSource Secure Acceptance Override Billing Address. |
| CsSAOverrideShippingAddress | CyberSource Secure Acceptance Override Shipping Address. |
| CsCvnDeclineFlags | CyberSource Ignore CVN Result (CVN). |

# Secure Acceptance Flex MicroForm

*Integration Overview*

Flex Microform provides the most secure method for tokenizing card data. Flex Microform is designed to be simple to integrate and allows you to focus on creating the best possible checkout experience for your customers. The provided JavaScript library enables you to replace the sensitive card number input field with a secure iFrame, hosted by CyberSource, that will capture data on your behalf. This embedded field will look and feel just like any other input in your checkout process allowing you to create a frictionless experience.

Once captured, the card number is replaced with a mathematically irreversible token that can only be used by you. The token can be used in place of the card number for follow on transactions in existing CyberSource APIs.

A Flex Microform consists of 2 main components:

1. A server-side component that requests limited use public keys from the Flex API
2. Using the Flex Microform client-side js library to seamlessly replace the sensitive PAN field in your input form.
3. The token created is used to do Credit Card Auth.

*Implementation*

To use CyberSource Flex Microform service on SFRA, ensure you have followed all steps in the "Cartridge Installation" guide above. A CyberSource Merchant ID, and CyberSource Merchant Key are required for this service. Enter these values in the corresponding site preferences under the "CyberSource" group. Follow the steps below to configure the service in Business Manager.

* 1. To complete the checkout process successfully, create API’s Key and shared secret on CyberSource business center console under 'Payment Configuration > Key Management Keys > Choose API Keys > Add Shared Secret Key'.
  2. Determine and set all associated site preference listed below.
  3. Set the ‘CsSAType’ site preference to ‘SA\_FLEX' to enable this service.

*Configuration*

Site Preference Group: CyberSource Secure Acceptance

|  |  |
| --- | --- |
| **Preference Name** | **Usage** |
| CsSAType | Secure Acceptance Type |
| SA\_Flex\_HostName | Flex Microform Host Name |
| SA\_Flex\_KeyID | Flex Microform Key ID |
| SA\_Flex\_SharedSecret | Flex Microform Shared Secret |

# Credit Card Capture Service

*Integration Overview*

This cartridge contains an interface that allows you to connect with the CyberSource Capture Service. There is no storefront connectivity with this interface, but it is available for you to use in your own integrations. CyberSource supports captures for all processors. When you are ready to fulfill a customer’s order and transfer funds from the customer’s bank to your bank, capture the authorization for that order. When fulfilling only part of a customer’s order, do not capture the full amount of the authorization. Capture only the cost of the items that you ship. When you ship the remaining items, request a new authorization, and then capture the new authorization.

A capture is a follow-on transaction that uses the request ID returned from a previous authorization. The request ID links the capture to the authorization. CyberSource uses the request ID to look up the customer’s billing and account information from the original authorization, so you are not required to include those fields in your capture request.

*Captures*

*Note: This section covers Capture service only for Credit Cards.*

Unlike authorizations, a capture does not happen in real time. All of the capture requests for a day are placed in a batch file and sent to the processor. In most cases, the batch is settled at night. It usually takes two to four days for your acquiring bank to deposit funds in your merchant bank account.

The following figure shows the steps that occur when you request a capture or credit.



1. You send a request for capture or credit over a secure Internet connection.
2. CyberSource validates the order information, then stores the capture or credit request in a

batch file.

1. After midnight, CyberSource sends the batch file to your payment processor.
2. The processor settles the capture or credit request and transfers funds to the appropriate

bank account.

*Implementation*

Ensure you have followed all steps in the "Cartridge Installation" guide above. A CyberSource Merchant ID, CyberSource Merchant Key are required for this service. Enter these values in the corresponding site preferences under the "CyberSource: Core" group.

The interface you will use to make capture requests is in the form of a single function:

CCCaptureRequest(requestID, merchantRefCode, paymentType, purchaseTotal, currency)

This function can be found in the script ‘scripts/facade/CardFacade.ds’. A working example of how to use this function can be found in the CYBServicesTesting-CaptureService controller. You will first get an instance of the CardFacade object, and make the call as follows:

var CardFacade = require('~/cartridge/scripts/facade/CardFacade');

var serviceResponse = CardFacade.CCCaptureRequest(requestID, merchantRefCode, paymentType, paymentTotal, currency);

The resulting serviceResponse object will contain the full response object generated by the request. The contents of this object will determine your logic in handling errors and successes. For detailed explanations of all possible fields and values, reference the Official CyberSource documentation for the CCCapture Service.

CCCaptureRequest parameter : Description

|  |  |
| --- | --- |
| **Parameter Name** | **Description** |
| requestID | Transaction ID obtained from the initial Authorization |
| merchantRefCode | SFCC Order Number |
| paymentType | Payment Type used for the Authorization |
| purchaseTotal | Order Total |
| currency | Cerrency code (ex. ‘USD’) |

# Customization

*Integration Overview*

The CyberSource SFRA cartridge has built-in custom hooks that can be utilized to customize request data being sent to each Service. This can be utilized to send additional custom data that the core cartridge cannot account for. For example, if you want to include Merchant Defined Data in your Credit Card Authorization Requests, you can use these hooks to achieve this.

The hooks are called in the ‘scripts/facade/CardFacade.ds’ and ‘scripts/facade/TaxFacade.js’ scripts. After a request for a particular service is built, but before it is sent to CS, a check for any code registering to the hook ‘app.cybersource.modifyrequest’ is done. If present, the hook will be called for that specific request. The request object is passed into the hook and the return value of the hook is sent to CS as the final request object. Through this process, you can inject your own data into the request object from custom code you write in a separate cartridge.

*Implementation*

To customize request objects, register the hook ‘app.cybersource.modifyrequest’ in your cartridges ‘hooks.json’ file. An example would look like this, replacing the script path with your own script :

{

"name": "app.cybersource.modifyrequest",

"script": "./cartridge/scripts/hooks/modifyRequestExample"

}

You can copy the ‘scripts/hooks/modifyRequestExample’ script from this cartridge into your own to use as a template for extending and modifying service request objects. Note, every hook must return a valid request object for the given service. It is recommended that you reference the CybserSource documentation for details on the exact nature of any fields you wish to customize or add. The following hooks are available for you to define in this file:

Modify Request hooks

|  |  |
| --- | --- |
| **Hook Name** | **Service Request to modify** |
| CCAuth | Credit Card Authorization |
| PayerAuthEnroll | Payer Authentication Enrollment |
| PayerAuthValidation | Payer Authentication Validation |
| AuthReversal | Credit Card Authorization Reversal |
| Capture | Credit Card Capture |
| Credit | Credit Card Credit/Refund |
| Tax | Tax Calculation |

**Release History**

|  |  |  |
| --- | --- | --- |
| **Version** | **Date** | **Changes** |
| 18.1.0 | 10-25-2018 | Initial SFRA release. |
| 18.1.1 | 11-6-2018 | Adds hooks to customize request objects.  Separates subscription creation option to use a new site preference.  Adds Facade for ‘Credit Card Credit’ Service.  Adds Facade for ‘Credit Card Capture Service.  Adds Facade for ‘Credit Card Auth Reversal’ Service. |