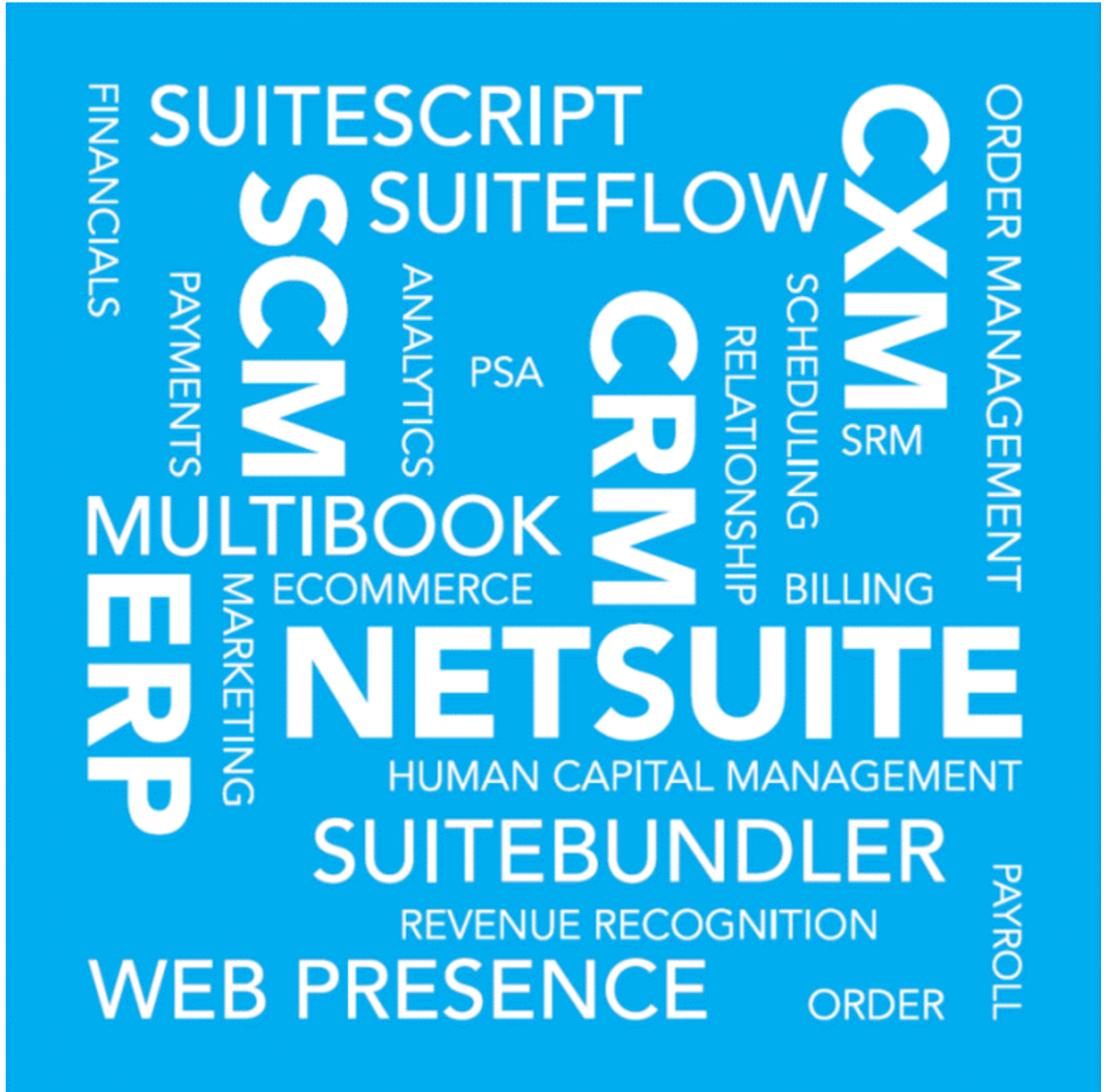


SuiteCommerce System Management



Copyright © 2005, 2018, Oracle and/or its affiliates. All rights reserved.

This software and related documentation are provided under a license agreement containing restrictions on use and disclosure and are protected by intellectual property laws. Except as expressly permitted in your license agreement or allowed by law, you may not use, copy, reproduce, translate, broadcast, modify, license, transmit, distribute, exhibit, perform, publish, or display any part, in any form, or by any means. Reverse engineering, disassembly, or decompilation of this software, unless required by law for interoperability, is prohibited.

The information contained herein is subject to change without notice and is not warranted to be error-free. If you find any errors, please report them to us in writing.

If this is software or related documentation that is delivered to the U.S. Government or anyone licensing it on behalf of the U.S. Government, then the following notice is applicable:

U.S. GOVERNMENT END USERS: Oracle programs, including any operating system, integrated software, any programs installed on the hardware, and/or documentation, delivered to U.S. Government end users are "commercial computer software" pursuant to the applicable Federal Acquisition Regulation and agency-specific supplemental regulations. As such, use, duplication, disclosure, modification, and adaptation of the programs, including any operating system, integrated software, any programs installed on the hardware, and/or documentation, shall be subject to license terms and license restrictions applicable to the programs. No other rights are granted to the U.S. Government.

This software or hardware is developed for general use in a variety of information management applications. It is not developed or intended for use in any inherently dangerous applications, including applications that may create a risk of personal injury. If you use this software or hardware in dangerous applications, then you shall be responsible to take all appropriate fail-safe, backup, redundancy, and other measures to ensure its safe use. Oracle Corporation and its affiliates disclaim any liability for any damages caused by use of this software or hardware in dangerous applications.

Oracle and Java are registered trademarks of Oracle and/or its affiliates. Other names may be trademarks of their respective owners.

Intel and Intel Xeon are trademarks or registered trademarks of Intel Corporation. All SPARC trademarks are used under license and are trademarks or registered trademarks of SPARC International, Inc. AMD, Opteron, the AMD logo, and the AMD Opteron logo are trademarks or registered trademarks of Advanced Micro Devices. UNIX is a registered trademark of The Open Group.

This software or hardware and documentation may provide access to or information about content, products, and services from third parties. Oracle Corporation and its affiliates are not responsible for and expressly disclaim all warranties of any kind with respect to third-party content, products, and services unless otherwise set forth in an applicable agreement between you and Oracle. Oracle Corporation and its affiliates will not be responsible for any loss, costs, or damages incurred due to your access to or use of third-party content, products, or services, except as set forth in an applicable agreement between you and Oracle.

If this document is in public or private pre-General Availability status:

This documentation is in pre-General Availability status and is intended for demonstration and preliminary use only. It may not be specific to the hardware on which you are using the software. Oracle Corporation and its affiliates are not responsible for and expressly disclaim all warranties of any kind with respect to this documentation and will not be responsible for any loss, costs, or damages incurred due to the use of this documentation.

If this document is in private pre-General Availability status:

The information contained in this document is for informational sharing purposes only and should be considered in your capacity as a customer advisory board member or pursuant to your pre-General Availability trial agreement only. It is not a commitment to deliver any material, code, or functionality,

and should not be relied upon in making purchasing decisions. The development, release, and timing of any features or functionality described in this document remains at the sole discretion of Oracle.

This document in any form, software or printed matter, contains proprietary information that is the exclusive property of Oracle. Your access to and use of this confidential material is subject to the terms and conditions of your Oracle Master Agreement, Oracle License and Services Agreement, Oracle PartnerNetwork Agreement, Oracle distribution agreement, or other license agreement which has been executed by you and Oracle and with which you agree to comply. This document and information contained herein may not be disclosed, copied, reproduced, or distributed to anyone outside Oracle without prior written consent of Oracle. This document is not part of your license agreement nor can it be incorporated into any contractual agreement with Oracle or its subsidiaries or affiliates.

For information about Oracle's commitment to accessibility, visit the Oracle Accessibility Program website at <http://www.oracle.com/pls/topic/lookup?ctx=acc&id=docacc>

Oracle customers that have purchased support have access to electronic support through My Oracle Support. For information, visit <http://www.oracle.com/pls/topic/lookup?ctx=acc&id=info> or visit <http://www.oracle.com/pls/topic/lookup?ctx=acc&id=trs> if you are hearing impaired.

Sample Code

Oracle may provide sample code in SuiteAnswers, the Help Center, User Guides, or elsewhere through help links. All such sample code is provided "as is" and "as available", for use only with an authorized NetSuite Service account, and is made available as a SuiteCloud Technology subject to the SuiteCloud Terms of Service at www.netsuite.com/tos.

Oracle may modify or remove sample code at any time without notice.

No Excessive Use of the Service

As the Service is a multi-tenant service offering on shared databases, Customer may not use the Service in excess of limits or thresholds that Oracle considers commercially reasonable for the Service. If Oracle reasonably concludes that a Customer's use is excessive and/or will cause immediate or ongoing performance issues for one or more of Oracle's other customers, Oracle may slow down or throttle Customer's excess use until such time that Customer's use stays within reasonable limits. If Customer's particular usage pattern requires a higher limit or threshold, then the Customer should procure a subscription to the Service that accommodates a higher limit and/or threshold that more effectively aligns with the Customer's actual usage pattern.

Beta Features

Oracle may make available to Customer certain features that are labeled "beta" that are not yet generally available. To use such features, Customer acknowledges and agrees that such beta features are subject to the terms and conditions accepted by Customer upon activation of the feature, or in the absence of such terms, subject to the limitations for the feature described in the User Guide and as follows: The beta feature is a prototype or beta version only and is not error or bug free and Customer agrees that it will use the beta feature carefully and will not use it in any way which might result in any loss, corruption or unauthorized access of or to its or any third party's property or information. Customer must promptly report to Oracle any defects, errors or other problems in beta features to support@netsuite.com or other designated contact for the specific beta feature. Oracle cannot guarantee the continued availability of such beta features and may substantially modify or cease providing such beta features without entitling Customer to any refund, credit, or other compensation. Oracle makes no representations or warranties regarding functionality or use of beta features and Oracle shall have no liability for any lost data, incomplete data, re-run time, inaccurate input, work delay, lost profits or adverse effect on the performance of the Service resulting from the use of beta features. Oracle's standard service levels, warranties and related commitments regarding the Service shall not apply to beta features and they may not be fully supported by Oracle's customer support. These limitations and exclusions shall apply until the date that Oracle at its sole option makes a beta feature generally available to its customers and partners as part of the Service without a "beta" label.

Domain Names

 **Applies to:** SuiteCommerce Web Stores

You can use your own domain name for your website, for web store checkout, and for email campaigns. For more information about using a custom domain for email campaigns, read the help topic [Campaign Email Domains](#).

Set Up Domain Names for your Website

Point your domain names at a NetSuite hosting domain to integrate your custom domains with your SuiteCommerce website.


For example, you have the following domain names registered: **shop.mysite.com** and **www.mysite.com**, and you want to use those domain names on the site you created using SuiteCommerce. First enter your domain names in NetSuite, and then configure Domain Name System (DNS) settings with your domain provider. Read the following help topics for more information:


- For general information about configuring domain names in NetSuite, read [Point Domain Names at NetSuite](#).
- For general instructions on setting up all types of domain names, read [Set Up Domains in NetSuite](#).

Set Up Secure Domains

In NetSuite, there are two different approaches to configuring your secure domains:

- You can choose from subdomains that are already configured in NetSuite. See [Set Up a Subdomain for Checkout and Shopping](#).
- You can register your own domains and purchase SSL certificates from the certificate authority of your choice. For a list of certificate authorities, see the [Mozilla Included CA Certificate List](#). To use a secure domain, contact your NetSuite account manager regarding access to the required features. For more information about secure checkout domain setup, read [Work with Custom Checkout Domains](#).

 **Note:** For a list of restrictions and recommendations regarding SSL certificates, see the topic [Purchase Domains and SSL Certificates](#).

 **Note:** You can purchase SSL certificates from providers not listed in the Mozilla Included CA Certificate list, however they may not be trusted by your selected web browser. Contact your certificate provider for more information.

Point Domain Names at NetSuite

To set up a domain for your website or email campaign, you must configure Domain Name System (DNS) settings to point your domain name at NetSuite website hosting servers. In most cases, your

domain provider can redirect your domain name using a CNAME record (Canonical Name record). This approach designates your domain name as an alias of the NetSuite hosting domains assigned to your account.



Important: Changing DNS configuration affects the ability to navigate to your site. It should be changed by someone with DNS setup experience, or consult with your domain provider.

To generate the alias you need to use for configuring a CNAME record, enter your registered domain name in NetSuite. For more information, see [Generate a CNAME \(Alias\) for a New Domain](#).



Warning: If you use the same domain name for both your website and for email hosting, you **cannot** use a CNAME record to point your website domain to NetSuite. You must set up an A record for your website domain. For more information, see [Website Domains and Email Hosting](#), in particular, the procedure on how to set up an A record.

Set Up a CNAME Record

Log into your domain provider website to manage DNS settings and to set up CNAME records. **This is a task that you must complete outside of NetSuite.**

Each domain provider has an online help system that gives detailed instructions for adding and editing CNAME records. Instructions vary depending on the domain provider. Make sure that you have copied the CNAME (Alias) displayed in your NetSuite account to use in DNS configuration settings with your domain provider.

Add or Change a Domain Name

Each time you add a new domain name in NetSuite, you must use the CNAME (Alias) displayed for that domain to configure DNS with your domain provider. If you change a domain name that you have entered in NetSuite, the corresponding CNAME (Alias) also changes. In this case, you must update DNS settings with your domain provider with the new CNAME (Alias).

Generate a CNAME (Alias) for a New Domain


The process of setting up a CNAME record for your SuiteCommerce website, or for campaign email involves entering your website domain in NetSuite, copying the CNAME (Alias) generated by NetSuite, and then configuring DNS settings with your domain name provider. Configuring DNS settings with your domain name provider is a task that you complete outside of NetSuite.

After entering your domain name in NetSuite, a CNAME (Alias) is immediately generated for that domain. To configure DNS settings, use the CNAME (Alias) generated by NetSuite to point your website domain name at NetSuite hosting servers. In NetSuite, you can view the CNAME (Alias) column on the Domains subtab of the Web Site Set Up page, and on the Set Up Domains page.


To obtain the CNAME (Alias) for a domain:

1. Purchase a domain from a domain name service provider and set up the domain. See [Set Up Domains in NetSuite](#).


If your domain is already set up, go to Setup > SuiteCommerce Advanced /Site Builder > Domains and click **Edit**.

 **Tip:** You can also open the domains list from the **Web Site Setup** page.

2. Copy the value in the **CNAME (Alias)** field.

 **Note:** The CNAME (Alias) assigned to each domain in each NetSuite Account is unique.


3. Visit your domain provider website to configure DNS using the CNAME provided by NetSuite. For more information, see [Set Up a CNAME Record](#).

 **Important:** If you use the same domain name for your website and for email hosting, you **cannot** use a CNAME record to point your website domain to NetSuite. You must set up an A record for your website domain. For more information, see [Website Domains and Email Hosting](#), in particular, the procedure on how to set up an A record.

Website Domains and Email Hosting


Standard best practices recommend that you should never use your domain for multiple purposes. The most common occurrence of this situation is when you are using your root domain to host several services. As an example, the website <https://wolfeelectronics.com> and the company email jwolfe@wolfeelectronics.com are sharing the same root domain.

To accomplish sharing a domain between shopping and email, you must to set up an A record for the domain with your domain provider. There are limitations imposed by the use of an A record, which ties your account to a specific IP address.

 **Warning:** If you are using the same root domain for your website and to setup email campaigns, you cannot use a CNAME to point your root domain to NetSuite shopping servers. Problems can occur when multiple DNS records are assigned to one root domain name. For example, if you set up **mydomain.com** in NetSuite for both shopping and email marketing, you may encounter problems such as email server failures.

You should be aware of the potential drawback of using the same root domain for both your website and your email services.

- Your account is not mobile:
 - When you are using an A record, you cannot use a CNAME(Alias) DNS record. Using a CNAME(Alias) DNS record is the only way to ensure that your account can be moved to (and your website hosted from) a different data center. Moving your account to a different data center might be required, for example, to better balance load across our infrastructure.
 - If your account is moved to a different data center, you must update the A record with the IP address for shopping that is appropriate for the new location hosting your site. Your shopping website might be offline until the A record has been updated.

 **Important:** If you are using an A record, you are responsible for updating your A record as soon as possible after your account has been moved to a different data center. See [If You Absolutely Must Use the Same Root Domain](#).

- You cannot use an A record for your shopping domain when using a Content Delivery Network, or CDN, for your website. If you use an A record for your website, the CDN you are using for your website is effectively bypassed. Your website is not able to reap the benefits that a CDN provides. This can mean slower page load times for your webstore, so using an A record can impose a performance penalty.


- Using an A record for a root domain is not supported for secure domains (URLs that start with `https://`, such as secure checkout domains).

However, under some circumstances, you might determine you have no other alternative than using an A record. If this is the case for your company, you must set up an A record with your DNS provider to accommodate hosting both your website and email on the same domain.


If You Absolutely Must Use the Same Root Domain

Set up an A record to point your website domain name to the IP address of the NetSuite shopping servers. Creating an A record for the root domain you plan to use for your shopping site is a task you must complete outside of NetSuite, with your domain provider. You must also enter your shopping domains on the Set Up Domains page in NetSuite.

For example, if you plan to use **mydomain.com** for both shopping and email, set up an A record for this domain with your domain provider. The A record must point to NetSuite shopping servers. Then, enter **mydomain.com** along with the subdomains you plan to use (such as, `www.mydomain.com`, `shop.mydomain.com`) in NetSuite on the Domain Setup page.

 **Note:** If you plan to use a certain domain only for email marketing, then set up a CNAME record for the campaign email domain. The A record is not required for domains used only for email marketing. For more information, read the help topic [Campaign Email Domains](#).

To set up an A record:

 **Important:** Only `http://` domains are supported when using an A record.

1. Log in to your domain provider website.
2. Follow instructions for setting up an A record.
3. Add a new A record if you do not already have an A record configured.
4. Replace the IP address provided by your domain provider with the appropriate IP address for the host data center for your account:
 - For `shopping.netsuite.com`, use 167.216.129.13
 - For `shopping.na1.netsuite.com`, use 64.89.45.13
 - For `shopping.na2.netsuite.com`, use 208.46.212.34
 - For `shopping.na3.netsuite.com`, use 72.34.169.17
 - For `shopping.eu1.netsuite.com`, use 185.72.129.17
 - For `shopping.eu2.netsuite.com`, use 212.25.248.17

For more information about domain URLs and multiple data centers, see the help topics [Understanding NetSuite URLs and Data Centers](#) and [Understanding Multiple Data Centers](#).

The A record now points to the correct IP address depending on your host NetSuite data center. For example:

Host	Points To
@	167.216.129.13

Set Up Domains in NetSuite

 **Applies to:** SuiteCommerce Web Stores


You can use the Set Up Domains page to add or edit domain details and to view the domain health status. Refer to the following sections for detailed information:

- [View Domain Health Status](#)
- [View, Add, or Edit Domain Names in NetSuite](#)
- [Add Domain Details](#)
- [Add Certificate Details](#)
- [Complete the Domain Setup](#)

View Domain Health Status

After you have set up a Domain, you can view its status on the Set Up Domains page where all your domains are listed. To help you troubleshoot domain setup issues, the Domain Health Status feature conducts health tests and flags any issues related to Domain Deployment, DNS, and Certificate.


You can view a summary of your domain's health on the Set Up Domains page by means of status icons displayed in the Deployment, DNS, and Certificate columns. If you observe any anomalies in these columns, click View next to the Domain you want to troubleshoot. The Domain Health Status section displays the cause and solution of the problem.

 **Note:** To actively monitor your domain health, add the Domain Configuration Alert reminder to your reminder portlet. For more information, see the help topic [Setting Up Reminders](#).

For information on the actions that can trigger domain setup errors and how to resolve them, see [Domain Setup Errors](#).

View, Add, or Edit Domain Names in NetSuite

1. Go to Setup > SuiteCommerce Advanced > Domains.

 **Tip:** You can also open the domains list or add a new domain from the **Web Site Setup** page.

2. To set up a new domain, click **New**.
To work with exiting domains, click **Edit** or **View**.

Add Domain Details

1. Select the **Protocol** type: Secure Domain or Non-Secure Domain.
2. Select how the domain will be **Hosted As**. Options vary based on whether the domain is Non-Secure or Secure.

Non-Secure Domain

- **Web Store** – Point the domain name to your SuiteCommerce shopping web site.
- **Not Active** – Park the domain name until you decide to use it.
- **Email Campaign** – Replace references to netsuite.com in your email and marketing templates with your own domain. (Required if you send more than 10,000 email per month with campaigns or email merge operations.)

- **Hosted Web Page** – Point the domain name to a page hosted from your NetSuite File Cabinet.
- **Promotional URL** – Redirect the URL to a promotional URL you create at Setup > Other Setup > Promotional URLs > New.
- **Redirect URL** – Redirect the domain to a URL on a different website or a page on a different site.


Secure Domain

- **Checkout** – Point the domain name to your secure SuiteCommerce checkout web site.
- **Secure Web Store** – Host your web store shopping under a secure domain.
SuiteCommerce Advanced only. See [Secure Shopping Domains](#) for more information.
- **Single Domain Web Store and Checkout** – Host your web store shopping and checkout under single, secure domain.
SuiteCommerce Advanced only. See [Single Secure Domain for Shopping and Checkout](#) for more information.

3. Enter the **Domain Name**.

The **CNAME (ALIAS)** populates automatically using the domain name.

Use the value in this field to configure DNS settings with your domain provider.

 **Note:** Since the Domain Name for a Secure Domain is generated from the certificate, the Domain Name field is dimmed for Secure Domains.


4. **Deployment Status** – changes automatically as the deployment and DNS propagation proceeds. Each status is prefixed with the deployment type: **NetSuite Hosting** or **CDN**.

- **Live** – Deployment successful, all processes complete.
- **Deployment in progress** – DNS propagation underway.
- **Removing** – DNS switched to new provider, old provider being removed.
- **Failed** – Deployment failed.

5. For **Web Site**, select the website for the domain name. (Required if you selected Hosted As > Web Store.)

6. For SuiteCommerce web store domains, check the **CDN** box. The option may not be available for all domain types.

See [CDN for Secure Checkout Domains](#) and [Configure CDN Caching](#).

 **Note:** CDN for secure checkout domains is available for SuiteCommerce Advanced only. Prior to choosing CDN for checkout, we recommend that you review the information about prerequisites, caching and IP address ranges. See [CDN for Secure Checkout Domains](#).

7. For **HTML Hosting Root**, select **Live Hosting Files**.


You can create Hosting Root folders other than Live Hosting Files. For example, if you are setting up a domain for a test site, select Staging. For more information about staging a Site Builder site, see the help topic [Website Staging Environment](#). For information about staging a SuiteCommerce site, see the help topic [Deploy to a NetSuite Sandbox](#).

8. (Optional) In the **Touch Points** field, you can set touch points for deploying SSP applications on your site. For more information, see [Linking Your Website to an SSP Application](#).


9. (Optional) Select a **Site Maintenance Domain**. For more information about setting up a custom site maintenance page and domain, see [Creating a Custom Site Maintenance Page](#).

10. (Optional) For **Not-Found Page**, select the optional hosted page where customers will be directed in the event they navigate to an inactive page. By default, customers are redirected to your home page.
Begin by entering the page name and then press Tab or click the down arrow for a list.
11. (Optional) If you selected Hosted As > Hosted Web Page, choose the **Home Page** file from your File Cabinet that you want the domain name to point to.
12. (Optional) For **Redirect URL** field, enter the URL where you want to redirect visitors on your site. (Required if you selected Hosted As > Redirect URL.)
13. If this is a secure domain, enter the Certificate Details in the next procedure. Otherwise, follow [Complete the Domain Setup](#).


Add Certificate Details

 **Warning:** Do not delete (disable) an existing certificate until the a domain certificate deploys. Otherwise you may have an interruption in service.

Complete these fields only if your domain is secure. Otherwise, follow the steps in [Complete the Domain Setup](#). Certificate Details are read-only for non-secure domains.

 **Note:** The certificate files are not validated or verified when they are uploaded to the file cabinet. You can upload files with any extension to the file cabinet. However, since 2017.1, the only files you will be able to select when you are setting up your domains in NetSuite are certificate files with the following extensions: .cer, .cert, .crt, or .pem. Files with extensions such as .pfx or .p12 will not work.

1. Select the **Certificate** SSL certificate file (.crt) that you uploaded to the File Cabinet to use with the secure domain.
Click the upload option next to the list if you need to upload a new certificate now.
Certificate Information – After you select the appropriate certificate and CA certificate files, this read-only field displays the secure domain name and the dates for which the SSL certificates are valid.
2. Select the **CA Certificate**. This is the intermediate certificate file (_ca.crt).
3. Enter the **Key Password** associated with your certificate key.
Key Password has a limit of 2,000 characters.
4. Enter the **Certificate Key** by copying data from the .key file and pasting into this field.
Your encrypted key is not exposed online when you return to the Domains Setup page. Because checkout is https://* only, you must provide the certificate and key.

 **Note:** The **Certificate Information** read-only field displays the secure domain name and the dates for which the SSL certificates are valid.

Complete the Domain Setup

1. Click **Save**.
NetSuite Hosting deployments take about 10 minutes. CDN deployments go live in up to 45 minutes. Use the **Deployment Status** to track progress.
2. Visit your domain provider's website to complete the task of setting up a CNAME record. (Required for website domains and campaign email domains.) For more information, see [Set Up a CNAME Record](#). See also, [Website Domains and Email Hosting](#).

After you enter your domain names in NetSuite and configured DNS with your domain provider, your domains begin redirecting. The time it takes for your domain to start redirecting to your website depends on your domain provider. Typically, redirection takes place from 2 – 48 hours after completing the setup.


Domain Setup Errors

This section describes the informational, warning, and error messages that you might see under Domain Health Status on the Set Up Domains page. You will also find information on the actions that can trigger these messages and solutions to resolve them. There are three types of messages:


- **Informational** – These messages provide information about an ongoing or completed task.
- **Warning** – These messages highlight exception conditions that might cause future problems if not resolved.
- **Error** – These messages indicate the task was not completed because of a user or system error and requires user action.


The issues fall into one of three categories:

- [Deployment](#)
- [DNS Record Setup](#)
- [Certification](#)

 **Note:** If the health status message is missing or uninformative, contact NetSuite Support.

Deployment

Message	Message Type	Cause	Action
Deployment successful	Informational	NA	NA
Live onDeployment to... in progress. <ul style="list-style-type: none"> ■ to CDN ■ to Netsuite 	Informational	NA	NA <div>  Note: NetSuite Hosting deployments take about 10 minutes. CDN deployments go live in up to 45 minutes. </div>
Live onRemoving from ... in progress. <ul style="list-style-type: none"> ■ from CDN ■ from NetSuite 	Informational	NA	NA
Deployment failed	Error	This issue seldom occurs. Some internal error might have caused this issue.	Contact NetSuite Support to resolve this issue.
Live on ... successful, deployment to ... failed.	Warning	This issue seldom occurs. Some internal error might have caused this issue.	Contact NetSuite Support to resolve this issue.
Deployment to ... in progress.	Informational	NA	NA

Message	Message Type	Cause	Action
			 Note: NetSuite Hosting deployments take about 10 minutes. CDN deployments go live in up to 45 minutes.
Removing from ... in progress.	Informational	DNS switched to new provider, old provider being removed.	NA

DNS Record Setup

Message	Message Type	Cause	Action
Your DNS record is configured correctly.	Informational	NA	NA
Your DNS record is configured incorrectly.	Error	You might have used your root domain to host several services causing this issue. For example, if you are using the same root domain for your website and to setup email campaigns, you cannot use a CNAME to point your root domain to NetSuite shopping servers. Problems can occur when multiple DNS records are assigned to one root domain name.	Do not use your root domain for multiple services. If you absolutely must use the same root domain for multiple services, follow the best practices described in Website Domains and Email Hosting .

Certification

Message	Message Type	Cause	Action
Your SSL certificate will expire soon.	Warning	NA	Renew your SSL certificate before it expires. For more information, see Purchase Domains and SSL Certificates .
Your SSL certificate has expired.	Error	NA	Renew your SSL certificate. For more information, see Purchase Domains and SSL Certificates .
Your CA certificate has expired.	Error	NA	Renew your CA certificate. For more information, see Purchase Domains and SSL Certificates .
Your certificate is issued by an untrusted CA.	Error	NA	Retrieve a certificate from any of the trusted CAs. For more information, see Purchase Domains and SSL Certificates .
Your certificate password is invalid.	Error	NA	Enter a valid certificate password.
Invalid certificate - Certificate format is not supported.	Error	<ul style="list-style-type: none"> You might have used unsupported certificate formats such as DER, P12, or P7B. 	Use a standard X.509 PEM-encoded certificate. Note: Do not use DER, P12, or P7B certificate formats as they are not supported.

Message	Message Type	Cause	Action
		<ul style="list-style-type: none"> You might have manually edited or merged the certificate files. 	
Your certificate key format is not supported.	Error	You might have used an unsupported PKCS#8 key format.	Use a PKCS#1 key format. Some certificate providers generate the private key encrypted in the unsupported PKCS#8 key format. For information on changing the private key from PKCS#8 to PKCS#1, see Secure Domains FAQ .
Invalid CA certificate - CA certificate format is not supported.	Error	<ul style="list-style-type: none"> You might have used unsupported certificate formats such as DER, P12, or P7B. You might have manually edited or merged the certificate files. 	Use a standard X.509 PEM-encoded certificate issued by a supported Certificate Authority. Note: Do not use DER, P12, or P7B certificate formats as they are not supported.
Invalid certificate - Wildcard certificates are not supported.	Error	NA	Use a single-domain SSL certificate. For more information, see Purchase Domains and SSL Certificates .
Self-signed certificate is not supported.	Error	NA	Obtain a certificate from any of the trusted CAs. For more information, see Purchase Domains and SSL Certificates .
Certificate issue detected.	Error	This issue seldom occurs. Some internal error might have caused this issue.	Contact NetSuite Support to resolve this issue.

Web Store Sessions

 **Applies to:** SuiteCommerce Web Stores

SuiteCommerce leverages two unique server environments, or domains, to support the eCommerce web store experience and maintain information related to current session:

- **Unsecured HTTP Shopper Domain:** supports the non-secure content and pages of the Shopping experience and related services.
- **Secure HTTPS Checkout Domain:** supports the secure content and pages of the Checkout and My Account experiences.



Note: NetSuite never sends credentials such as user names and passwords from an unsecured HTTP domain and always uses the secure HTTPS domain for authentication.

Both environments are deeply integrated into NetSuite and don't explicitly know the state or session information of each other. To achieve a seamless customer experience between the secure and unsecured domains, tokens and linkable attributes are passed between the two server environments via URL parameters and are stored as cookies to maintain the transferred state over time on each domain. This is commonly referred to as Domain Bridging.

Session Management

SuiteCommerce uses a combination of entities and roles to manage session information on and across domains.

Definitions:

- **Entity:** An entity is the identifier for a specific NetSuite user. An entity is typically of the type Customer, but can be other types including Vendor or Employee.
- **Roles:** A role is a defined access configuration within NetSuite. Roles are assigned to users and include sets of permissions for viewing and editing data. The Roles and associated permissions determine the pages that users can see and the tasks that they can complete.
- **Session:** A session is defined by the server environment to keep track of state associated with the current user experience within NetSuite. A session knows both the `EntityID` and the `Role` associated with the current user.

Roles

There are two NetSuite roles used in the context of Web Stores:

- [Shopper Role](#)
- [Customer Center Role](#)

Shopper Role

A shopper role indicates a user who is not authenticated and does not have a customer role. Anytime a user visits a web site, that user's session is assigned a shopper role by default when no other role has been established by a prior login in this session.

A Shopper role is a role in NetSuite that has no write permissions and read-only permissions to limited record types in the account, such as items. With a shopper role, users cannot take any action that would create a new record, other than registering as a customer.

Note: As a direct result of how the Shopper role is used in SuiteCommerce web stores, some functionality is not currently supported. For example, we currently do not support the newsletter functionality in SuiteCommerce web stores since a user with a Shopper role cannot directly create a new lead record.

Customer Center Role

The Customer Center role provides elevated permissions to records, such as transactions, that are required to create the Checkout and My Account experiences. This role can be customized to adjust the level of access. For example, you can remove links to transactions or records or limit access to only viewing instead of editing or creating records.

Session States

A Session State indicates the degree of authentication that the server understands about the current user. A Session State is managed throughout the life of the user's session and is determined by the combination of entity and role awareness.

SuiteCommerce Web Stores support three overall states:

- **Anonymous State** - A session is considered anonymous when the web store knows nothing about the user. The user does not have an `EntityID` defined. The anonymous session state typically occurs when someone comes to the site for the very first time, but can also occur after a customer has signed out. Anonymous users have full access to the unsecured HTTP Shopping domain. For example, `EntityID: 0 AND Role: Shopper`. Anonymous users can also have limited access to the secure HTTPS Checkout and My Account domains to access the ability to log in.
- **Authenticated State** - A user is considered authenticated when the user has a session that has been logged in. A logged in session means that the server environment knows both the `EntityID` of the user and the associated `Role`. For example, `EntityID: 1234 AND Role: Customer Center`.
- **Recognized State** - A user is considered recognized when the server environment knows the user's `EntityID` but that user does not have a valid session. For example, the user's role is Shopper instead of Customer Center. The recognized session state typically occurs when a customer who has logged in before returns to the site to shop again at a later time. `EntityID: 1234 AND Role: Shopper`.

State	EntityID	Role	Degree of authentication
Authenticated	1234	Customer Center	User is currently logged in
Recognized	1234	Shopper	User has logged in sometime in the past
Anonymous	0	Shopper	User is unknown, likely a first-time visit
IMPOSSIBLE	0	Customer Center	Authentication cannot occur

Sign In

If an anonymous user tries to access the secure HTTPS Checkout or My Account domains, they are required to log in or create an account, and see only a login page. If an anonymous user signs in

or creates an account on the secure HTTPS domain, their role is elevated from the Shopper Role to the Customer Center Role and an Entity ID is assigned. However, this is only for the secure HTTPS domain. After a login on the secure domain, the user's role and entity state has not yet changed on the unsecured HTTP domain. For the unsecured domain to be in sync, the user needs to navigate back to the unsecured domain, and through the link pass the appropriate information to sync the other session's information. After the unsecured domain is in sync, the user then has the same Entity ID (1234) and elevated Role (Customer Center) on both server environments.

Cookies

The following cookies are used to retain session specific information in a SuiteCommerce web store experience:

- **CkID/ShopperID** - These cookies link a specific browser to a specific cart. They reside in the browser and persists beyond the current browsing session. The values of these cookies are kept in sync between servers and determine how return customers are later recognized.
- **JSESSIONID** - This cookie links a specific browser to a specific session on one of NetSuite's servers. When the current shopping experience has spanned both the unsecured HTTP Shopping domain and the secure HTTPS Checkout or My Account domains, there is a different **JSESSIONID** associated with each domain.

Shopping Cart Use Case

When a user enters a web store for the first time, they are issued a ShopperID cookie that identifies the specific browser and computer being used, and references the Shopping Cart. When users navigate to checkout, the HTTP Shopping domain passes the ShopperID to the HTTPS Checkout Domain via a URL parameter called **ck**. This **ck** parameter identifies the user's browser and computer and its association with the user's cart. The user, browser, and cart connection between the two NetSuite server environments is preserved to create a single unified experience.

If a user later logs in to the same site from another computer or a different browser, a different **ck** parameter is passed. If the user then logs into the site from the new computer or browser, the preexisting cart is associated with the new **ck** parameter, as well as becoming associated with the user's Entity ID.

Session 1 (Work Computer)

A new shopper that has never visited the site goes to the web store. The user is not recognized and thus has an EntityID of 0 and the default Shopper role. At the same time, a shopping Cart is created (CartID: 10000) as well as a **ck** parameter which is associated with the newly created CartID. However, because the user has not created an account, the Entity ID for the shopping cart remains 0.

At this point, the user's session has the following attributes:

- Entity ID: 0
- Role: Shopper
- Shopping Cart Table
 - CartID: 10000
 - EntityID: 0
- Browser Table
 - - CartID: 10000
 - CkID: AAAAAAAAAA

Next, the user navigates to the secure domain, intending to log in or register. The CkID is passed from the shopping environment, and the checkout domain starts off with the same attributes as the shopping server. Because the session's role is Shopper, the user is presented a login/register page.

Now, when the user creates an account on the secure HTTPS Checkout domain, the shopping cart on the Checkout domain is associated with the newly created Entity ID resulting in a direct link with the shopping cart and the Customer. The user's session now has the following attributes:

- Entity ID: **1234** (Unique to this Customer)
- Role: Shopper
- Shopping Cart Table
 - CartID: 10000
 - EntityID: **1234**
- Browser Table
 - - CartID: 10000
 - CkID: AAAAAAAAAA

Session 2 (Home Computer)

The user leaves work and heads home. At home, they decide to go back to the web store to do some more browsing on their home computer. When the user goes back to the web store they are not recognized and are considered a new user again. The session has the following attributes:

- Entity ID: 0
- Role: Shopper
- Shopping Cart Table
 - CartID: 20000
 - EntityID: 0
- Browser Table
 - - CartID: 20000
 - CkID: BBBBBBBBBB

Next, the user navigates to the secure HTTPS Checkout domain, intending to log in again. The CkID is passed from the shopping environment, and the checkout domain starts off with the same attributes as the shopping server. Because the session's role is Shopper, the user is presented a login/register page.

At this point, the user decides to sign back into their account with the new credentials they created earlier at work. After the user logs in, the system recognizes an existing cart with the same Entity ID and updates the CkID to refer to the existing Cart ID. If the user added items on their home computer and the their cart was not empty from when they shopped at work, then the items added from home are merged together with the original cart. The user's session information is updated and now has the following attributes:

- Entity ID: **1234**
- Role: Customer Center
- Shopping Cart Table
 - CartID: 10000
 - EntityID:1234

- Browser Table
 - ■ CartID: 10000
 - CkID: BBBB BBBB



Note: Multiple CkIDs can be associated with a single cart. This is how NetSuite can synchronize carts between Work, Home and mobile devices.

Prerequisites for Setting Up Secure Domains

 **Applies to:** SuiteCommerce Web Stores



Important: On Site Builder websites using the latest version of Google Chrome browser, a security warning message might be displayed stating that the connection is not secure. Google has made modifications to Chrome to display the `Not secure` warning message if the website visitor enters data on an HTTP page outside Incognito mode, and on all HTTP pages visited in Incognito mode. The shopping pages on Site Builder websites use HTTP connections causing the warning message to be displayed. You can safely ignore this warning message as the Login and Checkout pages on Site Builder websites use HTTPS connections and experience no security risk.



Note: While it is possible to create secure HTTPS shopping domains using a third party Content Delivery Network (CDN) provider, this usage is not supported by NetSuite. Site Builder considers all shopping domains as HTTP and this can be incompatible with third party HTTPS shopping domains. For example, Site Builder always generates absolute URLs with HTTP prefix for files that are linked to using `<link href="">`. When these absolute HTTP URLs are used with third party HTTPS shopping domains, security warnings may be displayed.

Complete the steps below before you set up a custom domain in NetSuite:

1. [Purchase Domains and SSL Certificates](#)
2. [Download and Install OpenSSL](#)
3. [Create a Private Key for your Certificate](#)
4. [Generate a Certificate Signing Request \(CSR\)](#)
5. [Submit your CSR](#)
6. [Retrieve your Certificates](#)
7. [Upload your Certificate Files](#)



Note: You can purchase an SSL certificate from the certificate authority of your choice as long as it meets the restrictions listed in the topic [Purchase Domains and SSL Certificates](#). For a list of certificate authorities, see the [Mozilla Included CA Certificate List](#). You can purchase certificates from providers not listed in the Mozilla Included CA Certificate list, however, they may not be trusted by all web browsers.

Purchase Domains and SSL Certificates

You must complete all four tasks listed below **before** setting up your secure domain in NetSuite:

- **Choose a type of SSL certificate.**

For a list of certificate authorities, see the [Mozilla Included CA Certificate List](#).

You can select an SSL certificate from the vendor of your choice, but it must meet the following restrictions and recommendations:

- All SSL certificates you plan to use with NetSuite require:
 - A 2048-bit or 4096-bit RSA private key.

- The private key must use the PKCS#1 RSA Cryptography Standard.

Note: The PKCS#8 Private-Key Information Syntax Standard is not supported. See [Q:](#) if the private key issued to you uses the PKCS#8 standard.

- Must be Apache-compatible and PEM-encoded.
- It is recommended you purchase SSL certificates that use the SHA-2 hash function.
- The following are **not supported**:
 - Wildcard certificates
 - Self-signed certificates
 - ECC (Elliptic Curve Cryptography) SSL certificates
 - Subject Alternative Name (SAN) fields on an SSL certificate (that is, adding multiple domain names to a single certificate). Only the Subject Name on a certificate is considered. In cases where SANs are specified on a certificate (using a subjectAltName field), they are ignored.

Note: To test if a certificate is trusted by your selected web browser, click the link in the [URL to Test Website or Example Cert](#) column of the [Mozilla Included CA Certificate List](#). You can purchase certificates from providers not listed in the Mozilla Included CA Certificate list, however they may not be trusted by your browser. Contact your certificate provider for more information.

- **Contact your NetSuite account manager to ensure you have access to required features associated with Secure Domains.**

Certain fields that are required for the setup process in NetSuite do not appear unless you have enabled the required features. When you contact your account manager, specify how many secure domains you want to use. NetSuite allocates a CNAME (Alias), for each secure domain in your account.

- **Obtain a domain name to associate with the CNAME (Alias) provided by NetSuite.**

The domain name you use for checkout must be different from the domain name you use on the shopping pages of your web store. Consider using a domain such as `checkout.mycompany.com` or `store.mycompany.com`. Your secure domain name also appears on the My Account tab in the web store. Contact your domain provider to establish a new domain for web store checkout.

Important: The CNAME (Alias) generated by NetSuite is available only after the status of your custom checkout domain is Live.

- **Purchase SSL certificates.**

- You need SSL certificates for each checkout domain. Purchasing SSL certificates requires a separate set of steps that you complete outside of the NetSuite application. Contact the certificate authority that you are purchasing your certificate from for details.

Download and Install OpenSSL

OpenSSL is a utility that enables you to generate an SSL certificate and a private key. NetSuite requires certificates and private keys in the following contexts:

- When using custom checkout domains.

When using custom checkout domains, you must purchase an SSL certificate from a third party. See [Purchase Domains and SSL Certificates](#) for more information.

- When testing SuiteCommerce web stores using a secure domain on the local server.

Certificates and private keys are required when using HTTPS. You can generate the certificate and private key locally using OpenSSL. See the help topic [Secure HTTP \(HTTPS\) with the Local Server](#) for more information.

To download and install OpenSSL on Windows:

1. Download OpenSSL.


You can download binaries from [Shining Light Productions](#). Choose the correct version for your Windows system.

You can also download the OpenSSL source code and compile the binaries locally. See <http://www.openssl.org> for more information.

2. Configure the `OPENSSL_CONFIG` environment variable.

Set the `OPENSSL_CONFIG` environment variable in your path or on the command line using the following statement:

```
set OPENSSL_CONF=c:\OpenSSL-Win64\bin\openssl.cfg
```

 **Note:** Use the corresponding path for your installation.

To download and install OpenSSL on Other Platforms

On MacOS, use [Homebrew](#) to install OpenSSL. After installing Homebrew, use the following command:

```
brew install openssl
```

On Ubuntu Linux, use the following command to install OpenSSL:

```
apt-get install openssl
```

Create a Private Key for your Certificate

Use the OpenSSL utility to generate a private key. This is a file that contains a unique hash code. Keep this file stored on your computer. You will copy and paste the contents of this file into the enrollment form when you submit a CSR to your certificate authority, and later into NetSuite when setting up your secure domain. Your private key binds your certificate to your company.

To create a private key for your certificate:

1. Open a command prompt. Go to Start > Accessories > Command Prompt, or type CMD in the Run window.
2. Type `C:`, and then press Enter, to change the directory.
3. Type `openssl`, and then press Enter.
4. Type the following command:

```
genrsa -des3 -out <secure.domainnamekey>.key 2048
```

Note: Name the private key using the same domain name for which you are purchasing the certificate. For example, replace `<secure.domainnamekey>` with `store.wolfeelectronicskey`.

When you are prompted for a pass phrase, enter a secure password and remember it. Your pass phrase protects the private key. Both the private key and the certificate are required to enable SSL. You will create the certificate in the next step, [Generate a Certificate Signing Request \(CSR\)](#).

Generate a Certificate Signing Request (CSR)

Use the OpenSSL utility to generate the CSR required by your certificate authority. The CSR is used to verify your company's identity.

Important: You can purchase an SSL certificate from the certificate authority of your choice as long as it meets the restrictions listed in the topic [Purchase Domains and SSL Certificates](#). For a list of certificate authorities, see the [Mozilla Included CA Certificate List](#). You can purchase certificates from providers not listed in the Mozilla Included CA Certificate list, however, they may not be trusted by your selected web browser.

The numbered steps in this section provide general guidelines for generating the CSR, although each certificate authority may have specific requirements. Contact the certificate authority that you are purchasing your certificate from for details.

To generate the CSR:

1. Open a command prompt. Go to Start > Accessories > Command Prompt, or type CMD in the Run window.
2. Type `C:`, and then press Enter, to change the directory.
3. Type `openssl`, and then press Enter.
4. Type the following command:

```
req -new -key <secure.domainnamekey>.key -out <secure.domainnamecsr>.csr
```

Note: This command will prompt you to enter the following X.509 attributes for the certificate:

Country Name: Use the two-letter code without punctuation for country, for example: US for United States, or CA for Canada.

State or Province: Spell the state completely. Do not abbreviate the state or province name. For example, type California, not CA.

Locality or City: This is the city or town name, such as Berkeley, or Toronto. Do not abbreviate. For example, use Saint Louis not St. Louis.

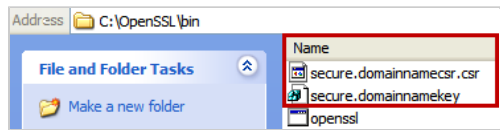
Company: Exclude any symbols from your company name. For example, XY & Z Corporation must be changed to XYZ Corporation, or XY and Z Corporation.

Organizational Unit (OU): (Optional). This field is used to identify certificates registered to a particular department or organization within a company. To skip the OU field, press Enter on the keyboard.

Common Name: This is the host and domain name, such as `secure.mydomain.com`, or `store.mydomain.com`.

After completing the steps above, you should have the following two files in **OpenSSL\bin** :

- A CSR file (.csr)
- A private key file (.key)



The private key (secure.domainname.key) is used for decryption. The CSR (secure.domainnamecsr.csr) is the public portion. You will submit data from both of these files to your certificate authority for enrollment.

Now that you have generated a private key and a certificate, you are prepared to [Submit your CSR](#).

Submit your CSR

After you select the type of SSL certificate you want to buy and you submit your payment information, the certificate authority typically requires that you submit a CSR and your private key.

Best practice indicates that you should have your private key and CSR ready when purchasing SSL certificates.

You must decide which type of SSL Certificate is most appropriate for your company. For a list of certificate authorities, see the [Mozilla Included CA Certificate List](#).

Note: You can purchase an SSL certificate from the certificate authority of your choice as long as it meets the restrictions listed in the topic [Purchase Domains and SSL Certificates](#).

Typically, the certificate authority requires you to provide your private key and CSR in the process of purchasing SSL certificates.

Submit a CSR

Copy and paste the data from the .csr file and the .key file into the appropriate areas of the enrollment form provided by the certificate authority. To do this, open each file in a text editor such as Notepad, and then save each of them as .txt files. Do not use Microsoft Word for this process, as it may insert extra hidden characters that alter the contents of the CSR.

Note specific instructions from your certificate authority regarding the characters that are not acceptable in the CSR (for example, < > ~ ! @ # \$ % ^ * / \ () ? , &).

After your company's identity has been verified, the certificate authority will typically send you an email message to inform you that it is time to [Retrieve your Certificates](#).

Retrieve your Certificates

After the certificate authority has validated your company's identity, they will notify you that your SSL certificates are ready for retrieval. The time it takes to get your SSL certificates signed varies depending on the type of certificate you request, and the certificate authority you choose to work with.

After you have successfully submitted your payment information and CSR, the certificate authority verifies your company's identity, and then issues your SSL Certificates. The process for retrieving

your certificates is slightly different for each vendor. Follow the directions provided by your certificate vendor to retrieve your certificate.

After Receiving Your SSL Certificates

After successfully retrieving your SSL certificates from the certificate authority, you should have a total of four certificate-related files on your computer:

- The certificate signing request file (.csr)
- The private key file (.key)
- The SSL certificate (for example, secure.domainname.crt)
- The intermediate certificate (for example, secure.domainname_ca.crt)


To complete the secure domain setup process in NetSuite, you only need to use the following three files:

- The private key file (.key)
- The SSL certificate (for example, secure.domainname.crt)
- The intermediate certificate (for example, secure.domainname_ca.crt)


Now, you can continue with the setup process and [Upload your Certificate Files](#).

Upload your Certificate Files

Upload your SSL certificate and your intermediate certificate file in the **SSL Certificates** folder in the NetSuite file cabinet.

 **Note:** The certificate files are not validated or verified when they are uploaded to the file cabinet. You can upload files with any extension to the file cabinet. However, since 2017.1, the only files you will be able to select when you are setting up your domains in NetSuite are certificate files with the following extensions: .cer, .cert, .crt, or .pem. Files with extensions such as .pfx or .p12 will not work.

- The SSL certificate (for example, secure.domainname.crt)
- The intermediate certificate (for example, secure.domainname_ca.crt)

 **Important:** Do not upload certificate keys to the file cabinet. You should store your certificate keys as securely as you would a password. You will copy data from this file in a later step.

You can upload multiple certificates in the **SSL Certificates** folder. If you are uploading a zipped file, unzip the file and upload the files individually. For more information about uploading files in NetSuite, see the help topic [Uploading Files to the File Cabinet](#).

After you have uploaded your certificate files, you can go to the next step, [Set Up Domains in NetSuite](#).

Secure Shopping Domains

 **Applies to:** SuiteCommerce Web Stores

This feature is available in the **Elbrus** release of SuiteCommerce and later. It is not applicable to Site Builder implementations.



Important: When working with implementations prior to the Elbrus release of SuiteCommerce, you **must** apply a patch when using HTTPS with shopping. For more information, see the help topic [Issue Patches](#).

You can use an SSL certificate to secure the shopping portion of your web store under an HTTPS domain. One option is to have a single, secure shopping domain paired with a single, secure checkout domain. Or you can have multiple shopping websites with a secure domain for each and then a separate single domain for checkout.

HTTPS is currently the industry standard for ecommerce services, and consumers prefer seeing the secure icon in their browser address bar. Including secure technology in your shopping area assures your customers that their activities on your site are safe. Also, search engines tend to rank secure sites higher than non-secure sites. Operating your web store shopping under a secure domain is one method for improving your search engine optimization (SEO).

Primary Domain Note: If you have a Secure Shopping Domain, only the Secure Shopping Domain can be set to Primary.


Set up Secure Shopping Domains

You can either switch from an existing non-secure shopping domain to a secure shopping domain or you can set up a new secure shopping domain. Switching from an existing non-secure shopping domain to a secure shopping domain is relatively easy, and the switch over is usually completed in about 45 minutes.

To switch from an existing non-secure shopping domain to a secure shopping domain:

1. Complete the prerequisite tasks for setting up secure domains as described in [Prerequisites for Setting Up Secure Domains](#).
2. If necessary, apply a patch.
Secure shopping domain is available in the **Elbrus** release of SuiteCommerce. If you are working with implementations prior to the Elbrus release, you **must** apply a patch as described in [Issue Patches](#).
3. Go to Setup > SuiteCommerce Advanced > Domains.
4. Click **Edit** for the desired shopping domain.
5. Select **Secure Domain** as the **Protocol** type.
6. From the **Hosted As** list, select **Secure Web Store**.
7. Add Certificate Details as described in [Add Certificate Details](#).
8. Click **Save**.

NetSuite Hosting deployments take about 10 minutes. CDN deployments go live in up to 45 minutes. Use the **Deployment Status** to track progress. The deployment is complete when the domain is in **Live** state.

 **Note:** You cannot make any changes to the domain settings until the deployment is complete.

In about 45 minutes, you will notice that the shopping domain address bar begins with https rather than http. This indicates that the switch over from an existing non-secure shopping domain to a secure shopping domain is complete.

To set up a new secure shopping domain:

1. Complete the prerequisite tasks for setting up secure domains as described in [Prerequisites for Setting Up Secure Domains](#).
2. If necessary, apply a patch.
Secure shopping domain is available in the **Elbrus** release of SuiteCommerce. If you are working with implementations prior to the Elbrus release, you **must** apply a patch as described in [Issue Patches](#).
3. Follow the steps under [Set Up Domains in NetSuite](#). Select **Secure Domain** as the **Protocol** type and under the **Hosted As** list, select **Secure Web Store**.

Test Secure Shopping Domains in a Safe and Isolated Environment

i Applies to: SuiteCommerce Web Stores

Secure shopping domain is available in the Elbrus release of SuiteCommerce, and if you are working with implementations prior to the Elbrus release, you must apply a patch.

i Note: Testing steps are different if you have a Sandbox provisioned for your Production account. Therefore, depending on your setup, you should either follow all the steps described in [Test on Accounts with Sandbox](#) or all the steps described in [Test on Accounts without Sandbox](#), but not both.

Test on Accounts with Sandbox

You can only test secure domains on a sandbox account if it resides on the system.netsuite.com domain.

i Note: If you are running your sandbox account on North American data centers, you can refresh your account and transition to the NetSuite domain. See the help topic [Sandbox Changes in North America](#) for information. If you have not refreshed your sandbox account to run on the NetSuite domain, you can test your non-secure domain setup in your Sandbox environment and then test your secure domain setup in your production environment.

Follow these steps to test the patch with your current implementation, and to test secure shopping domains:

1. If you are in a pre-Elbrus implementation, patch the SSP applications in your Sandbox environment as described in [Secure Shopping Domains \(Elbrus, Vinson, Mont Blanc, and Denali\)](#).
2. Create a new non-secure domain in your Sandbox environment as described in [Set Up Domains in NetSuite](#) and test it. Also, test an existing non-secure domain.

i Note: If you are using an EU Sandbox account, you can also set up a secure domain on sandbox for testing along with the non-secure domain.

3. If you are in a pre-Elbrus implementation, you must patch the SSP applications in your Production environment as described in [Secure Shopping Domains \(Elbrus, Vinson, Mont Blanc, and Denali\)](#).

If your Web Store is currently running on a customized bundle, it is strongly recommended that you follow these steps before applying the patch.

- a. Navigate to File Cabinet > Web Site Hosting Files > Live Hosting Files > SSP Applications.
- b. Make a copy of your development folder, which has all your customizations and store it in a new folder.
- c. Patch the bundle in the new folder as described in [Secure Shopping Domains \(Elbrus, Vinson, Mont Blanc, and Denali\)](#).
- d. Create a new SSP application by navigating to Setup > SuiteCommerce Advanced > SSP Applications > New. Ensure that all the fields in the new SSP application are same as the

SSP application of your current customized bundle. However, the new SSP application should point to the newly created folder and use a different value for the URL Root field.

4. Set up a new secure domain on production for testing.

Note: If you applied the patch, ensure that the touch points are pointing to the newly created SSP application while setting up the new secure domain.

If you get the following message when attempting to set up a new secure domain, request a testing subdomain on production from your NetSuite account representative:

You are not able to create an additional secure domain as all Secure Domain slots purchased for your account are in use.

Alternatively, you can free up an existing slot as described in [Free Up a Secure Domain Slot](#).

5. Test the secure domain on production.
6. Switch your main non-secure shopping domain on production to a secure shopping domain as described in [Set up Secure Shopping Domains](#).

Test on Accounts without Sandbox

Follow these steps to test the patch with your current implementation, and to test secure shopping domains:

1. If you are in a pre-Elbrus implementation, you must patch the SSP applications in your Production environment as described in [Secure Shopping Domains \(Elbrus, Vinson, Mont Blanc, and Denali\)](#).

If your Web Store is currently running on a customized bundle, it is strongly recommended that you follow these steps before applying the patch.

- a. Navigate to File Cabinet > Web Site Hosting Files > Live Hosting Files > SSP Applications.
- b. Make a copy of your development folder, which has all your customizations and store it in a new folder.
- c. Patch the bundle in the new folder as described in [Secure Shopping Domains \(Elbrus, Vinson, Mont Blanc, and Denali\)](#).
- d. Create a new SSP application by navigating to Setup > SuiteCommerce Advanced > SSP Applications > New. Ensure that all the fields in the new SSP application are same as the SSP application of your current customized bundle. However, the new SSP application should point to the newly created folder and use a different value for the URL Root field.

2. Set up a new secure domain on Production for testing.

Note: If you applied the patch, ensure that the touch points are pointing to the newly created SSP application while setting up the new secure domain.

If you get the following message when attempting to set up a new secure domain, request a testing subdomain on production from your NetSuite account representative:

You are not able to create an additional secure domain as all Secure Domain slots purchased for your account are in use.

Alternatively, you can free up an existing slot as described in [Free Up a Secure Domain Slot](#).

3. Test the secure domain in production.
4. Switch your main non-secure shopping domain on production to a secure shopping domain as described in [Set up Secure Shopping Domains](#).

Free Up a Secure Domain Slot

If you are already using all the available secure domain slots, you can free up a slot in use so that you can set up a secure shopping domain or a single domain for shopping and checkout. You can achieve this by replacing the Custom Checkout Domain with a Preset Domain.

Prerequisites

- The Customizable Checkout Subdomains feature should be enabled in your account.
 1. Navigate to Setup > Company > Enable features.
 2. Click the **Web Presence** subtab.
 3. Enable **Customizable Checkout Subdomains** if already not enabled.
- If any of your customized scripts or functionalities have the checkout domain name hardcoded in them, ensure that you edit them beforehand. They might stop functioning as soon as you make the following change.

To set up a preset checkout domain instead of an existing custom secure domain:

1. Navigate to Setup > SuiteCommerce Advanced > Set Up Web Site.
2. Click **Edit** for the desired website.
3. Click the **Domains** subtab.
4. In the **URL Alias** field, enter the URL alias for your preset checkout domain.
For example, enter mystore in the URL Alias field to use mystore as the prefix for your checkout domain.
5. From the **Checkout URL** dropdown, select one of the following preset domains: securedcheckout.com, onlinecheckout.com, webstorecheckout.com, or suitecheckout.com.
6. Click **Save**.

If you have entered mystore in Step 4 and have selected securedcheckout.com in Step 5, your checkout URL will look similar to the following: `mystore.securedcheckout.com`.

Now that you have a free secure domain slot available, you can use this slot to set up [Secure Shopping Domains](#) or [Single Secure Domain for Shopping and Checkout](#).

Single Secure Domain for Shopping and Checkout


 **Applies to:** SuiteCommerce Web Stores

This feature is available in the **Elbrus** release of SuiteCommerce and later. It is not applicable to Site Builder implementations.

You can run your shopping and checkout under a single HTTPS domain. A single domain provides your customers with a seamless buying experience. It removes the delays associated with shopping in one domain and moving to another for checkout.

Primary Domain Notes:

- If your site includes a Single Domain setup, only the Single Domain can be set to Primary.
- If your site includes multiple Single Domains, no domain can be set to Primary.
- If your site includes at least one secure domain, you cannot set a non-secure domain to Primary.

 **Note:** This feature may be backportable to previous releases of SuiteCommerce Advanced. For more information, see the help topic [Issue Patches](#).

Set up Single Secure Domain for Shopping and Checkout


To set up a new single, secure domain for both web store shopping and checkout, follow the steps under [Set Up Domains in NetSuite](#). Select **Secure Domain** as the **Protocol** type and under the **Hosted As** list, select **Single Domain Web Store and Checkout**.

If you are already using two separate domains for shopping and checkout, you can switch to a single secure domain by either creating a new single secure domain or by upgrading an existing shopping domain to a single secure domain.

To upgrade an existing shopping domain to a single secure domain:

1. Verify the domain configuration of your shopping domain.
If the domain configuration indicates that your shopping domain is a non-secure domain, you should switch the existing non-secure shopping domain to a secure shopping domain. To accomplish this, follow the steps under [Set up Secure Shopping Domains](#).
2. Once the shopping domain is secure, go to Setup > SuiteCommerce Advanced > Domains.
3. Select the secure shopping domain to be upgraded to a single secure domain, and click **Edit**.
4. From the **Hosted As** list, select **Single Domain Web Store and Checkout**.
5. Click **Save**.

NetSuite Hosting deployments take about 10 minutes. CDN deployments go live in up to 45 minutes. Use the **Deployment Status** to track progress.

 **Note:** You cannot make any changes to the domain settings until the deployment is complete.

In about 45 minutes, both your web store shopping and checkout applications will be running under a single HTTPS domain.

Set Up a Subdomain for Checkout and Shopping

 **Applies to:** SuiteCommerce Web Stores

Customers on your SuiteCommerce web site enter payment information and submit orders on a secure checkout server. In addition, you can set up a secure domain for shopping.

For checkout, when customers check out, they are automatically redirected to a secure checkout domain hosted by NetSuite: checkout.netsuite.com. You can choose to suppress NetSuite branding on the checkout domain, and use one of the secure subdomains, that are already configured by NetSuite:

- securedcheckout.com
- onlinecheckout.com
- webstorecheckout.com
- suitecheckout.com

Whether you choose to use one of the checkout subdomains, the default NetSuite checkout subdomain, or a secure shopping subdomain, shoppers on your web site and browse and enter their information using a Secure Sockets Layer (SSL) connection. An SSL connection is protected by encryption. Shoppers know they have an SSL connection because the browser displays a gold padlock and the address bar begins with https rather than http. SSL provides the confidence that private data sent to the web site is kept confidential.

Extended Validation and Wildcard SSL

The NetSuite default checkout subdomain (checkout.netsuite.com) uses Extended Validation (EV) a special type of certificate which requires more extensive investigation for validating an entity before the certificate is issued.

The checkout subdomains use Wildcard SSL Certificates to enable SSL encryption. This means that multiple host names (or subdomains) are covered by a single SSL Certificate. Wildcard certificates currently do not support EV, so the checkout subdomains do not support EV.

You also have the option to create your own custom secure domain to use for web store checkout. Setting up your own secure checkout domain requires extended setup steps and incurs additional cost. For more information, see [Work with Custom Checkout Domains](#).

To set up a secure subdomain for checkout:

Select the subdomain on the Set Up Web Site page. NetSuite uses your web store alias to construct a unique checkout subdomain for your site.

1. Go to Setup > Company > Enable Features. Enable the required feature.
 - a. Click the **Web Presence** subtab.
 - b. Check the box next to **Customizable Checkout Subdomains**.
 - c. Click **Save**.
2. Go to Setup > SuiteCommerce Advanced > Set Up Web Site.
3. Click **Edit** for the desired web site.
4. On the **Domain** subtab, enter your checkout **URL Alias**. Enter the alias as one word, all lower case and without punctuation.

The alias is appended to the subdomain to construct your checkout URL. An alias can make it easier for a shopper to identify areas on your site.

Checkout Domain		
Edit View	<div>URL ALIAS</div> <div>mystore</div>	<div>CHECKOUT URL HTTPS://MYSTORE.</div> <div>onlinecheckout.com</div>

5. Select your **Checkout URL**.
6. Click **Save**.

After you set up a checkout subdomain, you may experience a delay in redirection of up to 30 minutes if you choose to change from one checkout subdomain to another. The delay is due to the invalidation of the previous subdomain in the web store caches.

To set up a secure shopping subdomain:

1. See [Secure Shopping Domains](#), and then follow the steps linked there to [Set Up Domains in NetSuite](#).
2. Select **Secure Web Store** from the **Hosted As** list.

Work with Custom Checkout Domains

 **Applies to:** SuiteCommerce Web Stores

A custom checkout domain lets you maintain your company branding throughout a shopper's web store experience, including checkout and the customer center.

For example, when you operate a web store with NetSuite, by default, shoppers submit orders on the checkout domain, <https://checkout.netsuite.com>. Rather than use the default NetSuite checkout domain, you can use your company's web site domain in the checkout URL. Your custom checkout domain is also visible when customers visit the My Account tab in your web store.

Use a Custom Checkout Domain

To implement a custom checkout domain in your web store, you must use two domains. The checkout domain must be different from the shopping domain on your web site. For example, if you use <http://www.mycompany.com> for the shopping section of your web site, you might use <https://checkout.mycompany.com> for checkout.


Checkout domains require SSL certificates. You are responsible for obtaining the certificates and determining which type of certificate is appropriate for your web site. You are also responsible for renewing and maintaining your SSL certificates, as well as configuring the DNS for your checkout domain.

For more information, see [Certificate Maintenance](#).


Use SSL Certificates with a NetSuite Web Store

Purchasing Secure Socket Layer (SSL) certificates for your checkout domain lets you control the domain displayed in the address bar on web store checkout pages.

SSL certificates are credentials issued for a specific domain and web server, authenticated by a Certificate Authority (CA), or SSL certificates provider. This digital certificate binds a domain and a server's identity to a pair of electronic keys.

 **Note:** You can purchase an SSL certificate from the certificate authority of your choice as long as it meets the restrictions listed in the topic [Purchase Domains and SSL Certificates](#). For a list of certificate authorities, see the [Mozilla Included CA Certificate List](#). You can purchase certificates from providers not listed in the Mozilla Included CA Certificate list, however, they may not be trusted by all web browsers.

In the same way that checkout for your web store is secure on <https://checkout.netsuite.com>, your custom SSL certificates help to ensure private communication over the public internet by encrypting information transferred, in this case, during ecommerce transactions.

 **Note:** You also need to generate a self-signed SSL certificate to test a secure domain using HTTPS **locally** during development of a SuiteCommerce web store. For details, see the help topic [Secure HTTP \(HTTPS\) with the Local Server](#).

NetSuite allocates a CNAME for your checkout domain through which it hosts your certificate files.

Certificate Maintenance

You are responsible for renewing and maintaining your SSL certificates with your certificate provider, as well as configuring DNS for your checkout domain with your domain provider. Maintaining your certificates also includes updating information in NetSuite. NetSuite is responsible for allocating a CNAME for your checkout domain, and binding your SSL certificates to that CNAME.

Note: This section is about maintenance tasks associated with your checkout domain. For information on purchasing domains and certificates, creating certificate keys, and uploading certificates, see [Prerequisites for Setting Up Secure Domains](#).

You are responsible for the following maintenance tasks associated with your checkout domain:

- Renewal
- Update Certificates in NetSuite
- Replacement
- Upgrade
- Revocation

Renewal

SSL certificates are valid for a certain period of time. You are responsible for keeping track of the date when it is time to renew your certificate with your certificate provider. NetSuite displays the valid dates for your certificate on your domain records.

To view valid dates for an SSL certificate:

1. Go to Setup > SuiteCommerce Advanced /Site Builder > Domains.
2. Click **View** for the desired domain.
3. Review the **Certificate Information** field.

Contact your SSL certificate provider for more information about each of the maintenance tasks associated with your certificates.

Update Certificates in NetSuite

After you complete each of the maintenance tasks with your certificate provider, you must update the certificate information in NetSuite. Adding, deleting, and updating files in the file cabinet have no effect on certificates deployed with NetSuite. These actions only affect which certificates are available to select on the Domains Setup page.

For example, when you renew the SSL certificates for your secure domain, you can update your certificates in the file cabinet, but no changes are initiated until you select the updated files on the Domains Setup page.

Warning: Some certificate vendors, such as GoDaddy, may revoke an old certificate 72 hours after issuing a new one.

To update your certificate files in NetSuite:

1. Upload or edit the certificate files in the **SSL Certificates** folder in the file cabinet.

2. Go to Setup > SuiteCommerce Advanced /Site Builder > Domains.
3. Click **Edit** for the desired domain.
4. Select the updated files in the **Certificate** field and the **CA Certificate** field.
5. In the **Certificate Key** field, enter the updated key.
6. In the **Key Password** field, enter the password, if applicable.
7. Click **Save**.

Your existing secure domain is continuously served until the new certificate is deployed. NetSuite deploys domains often during the day. During this time, checkout through your secure domain continues uninterrupted. You will get a notification email message when the new certificate is Live.

Replacement

You may need to replace your SSL certificates when setting up your NetSuite web store, or in the event that your private key is lost, or becomes corrupted. Alternatively, you may need to update your certificate to change the domain name. In this case, you must use the default checkout URL, **https://checkout.netsuite.com**, until your new custom checkout domain is live.

Upgrade

You may be allowed to upgrade your SSL certificates based on the options provided by your certificate authority.

Revocation

You may need to revoke your certificate if it becomes compromised in any way.

Secure Domains FAQ


How do I generate a Certificate Signing Request (CSR)?

To find the information you need from NetSuite, see [Prerequisites for Setting Up Secure Domains](#). Next, follow the instructions for generating a CSR posted on your certificate authority's web site. If you cannot find these instructions, contact your certificate authority for details.

What are the requirements for the SSL certificates I buy for the custom checkout domain I use with NetSuite?

You can select an SSL certificate from the vendor of your choice, but it must meet the following restrictions and recommendations:

- All SSL certificates you plan to use with NetSuite require:
 - A 2048-bit or 4096-bit RSA private key.
 - The private key must use the PKCS#1 RSA Cryptography Standard

 **Note:** The PKCS#8 Private-Key Information Syntax Standard is not supported. See [Q:](#) if the private key issued to you uses the PKCS#8 standard.

- Must be Apache-compatible and PEM-encoded.

- It is recommended you purchase SSL certificates that use the SHA-2 hash function.
- The following are **not supported**:
 - Wildcard certificates
 - Self-signed certificates
 - ECC (Elliptic Curve Cryptography) SSL certificates
 - Subject Alternative Name (SAN) fields on an SSL certificate (that is, adding multiple domain names to a single certificate). Only the Subject Name on a certificate is considered. In cases where SANs are specified on a certificate (using a subjectAltName field), they are ignored.

Note: To test if a certificate is trusted by your selected web browser, click the link in the **URL to Test Website or Example Cert** column of the [Mozilla Included CA Certificate List](#). You can purchase certificates from providers not listed in the Mozilla Included CA Certificate list, however they may not be trusted by your browser. Contact your certificate provider for more information.

How can I change the private key from PKCS#8 to PKCS#1?

Some certificate providers generate the private key encrypted in the unsupported PKCS#8 key format. The unsupported PKCS#8 key starts with the following line:

```
-----BEGIN PRIVATE KEY-----
```

You can convert this unsupported PKCS#8 key to the PKCS#1 key format using the following command:

```
$ openssl rsa -in <my-key-filename>.key -out <my-key-filename>-rsa.key
```

The PKCS#1 formatted key starts with the following line:

```
-----BEGIN RSA PRIVATE KEY-----
```

Can I test my secure domain on Sandbox or Release Preview?

Customers with sandbox accounts on the NetSuite domain and all Release Preview accounts can deploy custom secure domains. You must create a unique domain for your sandbox or Release Preview account. For example, if you use `https://checkout.mycompany.com` in your production account, you could use `https://test.checkout.sandbox.mycompany.com` or `https://test.checkout.releasepreview.mycompany.com`.

Note: Do not attempt to reuse domains that are already deployed in your production account. You must set up a unique domain for each of your accounts: production, sandbox, and Release Preview.

Note: If you are running your sandbox account on North American data centers, you can refresh your account and transition to the NetSuite domain. See the help topic [Sandbox Changes in North America](#) for information. If you have not refreshed your sandbox account to run on the NetSuite domain, you can test your non-secure domain setup in your Sandbox environment and then test your secure domain setup in your production environment.

Follow the instructions for setting up domains as you would for your production account. See [Prerequisites for Setting Up Secure Domains](#).

Can I delete my secure domain?

Yes.

Maintain Web Site Domains

i Applies to: SuiteCommerce Web Stores

Use the Domains subtab on the Web Site Setup page to view the checkout and shopping domains that you created for your site. The subtab includes options for adding and working your domains, as well as the option to make a domain your web site's Primary Web Site URL.

i Note: Only shopping and checkout domains are shown on the Web Site Setup > Domains subtab. To manage domains used for campaign email, redirects, and secure checkout, go to Setup > SuiteCommerce Advanced > Domains. For more information, see [Set Up Domains in NetSuite](#).

Setup Shopping Analytics Search Index Field Sets Email Upsell Advanced Touch Points **Domains** System Notes

Enter your custom domain in the **Domain Name** column. To use a custom domain with NetSuite, you must set up a DNS entry with your domain provider. The **CNAME (Alias)** column shows the NetSuite domain you must point to in the DNS entry.

Checkout Domain

Edit | View URL ALIAS: mystore CHECKOUT URL: HTTPS://MYSTORE.onlinecheckout.com

Shopping Domains

Add New Domain Manage Domains

EDIT	VIEW	DOMAIN NAME	PRIMARY WEB SITE URL	CDN	CHECKOUT DOMAIN
Edit View		http://advanced.localhostloopback.com	<input type="checkbox"/>	No	https://se4.eng.netsuite.com

Save Cancel Reset New Web Site Actions

To view and manage Checkout and Shopping Domains for your Web Site:

- Go to Setup > SuiteCommerce Advanced /Site Builder > Set Up Web Site.
- Click **Edit** for the desired website.
- Click the **Domains** subtab.
- Work with your **Checkout Domain**:
 - Edit or View** – Click to manage or display details about your current checkout domain.
 - URL Alias** – Enter the alias as one word, all lower case and without punctuation. The alias is appended to the subdomain to construct your checkout URL. An alias can make it easier for a shopper to identify areas on your site.
 - Checkout URL** – Select the checkout domain for your website.
- Work with your **Shopping Domain**:
 - Edit or View** – Click to manage or display details about a shopping domain.
 - Domain Name** – The name of the shopping domain.
 - Primary Web Site URL** – Select to redirect any domain you set up in NetSuite to this domain. (Required for Site Builder websites.) See [Select a Primary Web Site URL](#).
 - CDN** – Displays if the domain is using a Content Delivery Network (CDN).
 - Checkout Domain** – The checkout domain associated with the shopping domain.
 - Add New Domain** – Click to set up a new domain. See [Set Up Domains in NetSuite](#)
 - Manage Domains** – Click to display a list of your current domains.
- Click **Save**.

Select a Primary Web Site URL

Selecting a Primary Web Site URL is optional for SuiteCommerce websites, but required for Site Builder. Note that when you designate a Primary Web Site URL, any domain you set up in NetSuite redirects to the domain you marked as the Primary Web Site URL. The domain name marked as the Primary Web Site URL shows on your list of websites as the Primary Domain.

For example, you might set up two shopping domains such as **www.mystore.com** and **shop.mystore.com**. If you set **www.mystore.com** as the primary domain, then customers who type **shop.mystore.com/item** in the browser will be redirected to the same page on the primary domain, **www.mystore.com/item**.

Sticky Domains

In SuiteCommerce, if you do not designate a primary domain, then each domain name persists as shoppers navigate around your site. You can modify the shopping experience for customers by associating different domains with different SSP applications.

Best Practice for Using Sticky Domains

You can use an SSP application along with multiple shopping domains in one SuiteCommerce site. In this case, you can use the Touch Points subtab to associate an SSP application with a touch point for the site. Next, use the Domains setup to associate a domain with an SSP application.

Touch Points

Touch points defined on a domain override the touch points defined on the **Touch Points** subtab on the **Web Site Setup** page. For information on using touch points, see [Linking Your Website to an SSP Application](#).

Creating a Custom Site Maintenance Page

 **Applies to:** SuiteCommerce Web Stores

You can design and upload a custom maintenance page to display while you complete routine updates and during NetSuite scheduled maintenance periods. By designing your own maintenance page, you can continue to communicate your brand message even when your site is offline.



Important: Maintenance domains and displaying a custom site maintenance page are not supported on Sandbox.

To set up the maintenance page, you must configure your own site maintenance domain name with your DNS provider. Next, upload assets for your site maintenance page in the file cabinet in NetSuite. Then, create the maintenance domain and designate the maintenance folder. Finally, on the domains page in NetSuite, associate the site maintenance domain with a web site domain.

All files in the folder that you use to store your maintenance page are publicly accessible using the site maintenance domain. Note that SSP files and touch points are not supported. Also, SuiteScript is not executed on files in the site maintenance folder. If you use multiple web sites, you can create a different site maintenance domain for each web site domain.

To set up a custom site maintenance HTML page:

1. Create a domain name to use when your site is offline for maintenance, for example, **maintenance.mywebsite.com**
 - a. Configure DNS settings with your domain provider.
 - b. Point your site maintenance domain to the same NetSuite CNAME you use for your web site domains. For more information, see [Generate a CNAME \(Alias\) for a New Domain](#).
2. In NetSuite, create a subfolder in the **Web Site Hosting Files** folder to host the assets for your maintenance page. There is a 10 MB size limit to the folder you use for hosting your site maintenance page.
3. Upload the assets for your maintenance page in the folder you created. Your assets must include a file named **index.html**.
4. Go to Setup > SuiteCommerce Advanced > Site Maintenance Domains > New.
 - a. Enter your **Site Maintenance Domain Name**. This is the domain name you configured in Step 1.
 - b. Select a **Site Maintenance Folder**. This is the folder you created in Step 2.
 - c. Click **Save**.
5. Use the Domains page in NetSuite to map the site maintenance domain to your web site shopping domain.
 - a. Go to Setup > SuiteCommerce Advanced > Domains > New.
 - b. Click the web site domain name you want to associate with the site maintenance domain, then select a domain in the **Site Maintenance Domain** column. You can select a different site maintenance domain for each web site domain.
 - c. Click **OK**.
 - d. Click **Save**.

After you enter your new site maintenance domain in NetSuite (Step 4), it can take from 5 minutes up to 30 minutes to display your content online. To preview or test your maintenance page, open a

new browser and navigate directly to the site maintenance domain. For example, to preview your maintenance page, go to <http://maintenance.mywebsite.com>.

Note: Any time you change the maintenance domain, the Site Maintenance Folder or its contents, there is a lag time of about five minutes before you can see changes to the page.

Using the Site Maintenance Domains List

You can view a list of all the maintenance domains set up in your account by going to Setup > Site Builder > Site Maintenance Domains > New, or Setup > SuiteCommerce Advanced > Site Maintenance Domains.

From the list page, you can change the Site Maintenance Folder for a domain. You cannot change a site maintenance domain name after you have entered it in NetSuite. If you make a mistake in entering the site maintenance domain name, you must enter a new one correctly. You can delete a site maintenance domain after you remove the site maintenance domain from the Domains page.

To edit the maintenance domain folder:

1. Go to Setup > Site Builder > Site Maintenance Domains > New, or Setup > SuiteCommerce Advanced > Site Maintenance Domains > New.
2. Click **Edit** next to a site maintenance domain.
3. Select a different folder from the **Site Maintenance Folder** list.
4. After making your changes, click **Save**.

After a site maintenance domain has been associated with a web site domain name, it must be removed from the domains list, before it can be deleted.

To delete a maintenance domain:

1. Go to Setup > Site Builder > Domains, or Setup > SuiteCommerce Advanced > Domains.
 - a. Find the Site Maintenance Domain you want to delete.
 - b. Click on the **Domain Name** to which the site maintenance domain corresponds.
 - c. Select the blank entry to remove the maintenance domain from the list.
 - d. Click **OK**.
 - e. Click **Save**.
2. Go to Setup > Site Builder > Site Maintenance Domains, or Setup > SuiteCommerce Advanced > Site Maintenance Domains.
 - a. Click **Edit** next to the Site Maintenance Domain you want to delete.
 - b. On the Set Up Site Maintenance page, click the **Actions** menu.
 - c. Select **Delete**.
 - d. Click **Save**.

Taking your Web Site Offline for Maintenance


You may want to take your web site offline while you make changes such as updating images, adding banners, or editing product descriptions. When you take your site offline for maintenance, your custom

site maintenance page is displayed. If you do not create your own custom maintenance page, NetSuite generates a basic message to show on your site. The message automatically generated by NetSuite includes your web site display name. If you have uploaded a logo, then the logo displays along with the message, "<Web site display name> is temporarily offline for maintenance. Please try again later."

To take your web site offline for maintenance:

1. Ensure you have set up a custom site maintenance HTML page hosted on a site maintenance domain. Refer to the detailed instructions in [Creating a Custom Site Maintenance Page](#).
2. Go to Setup > Site Builder > Set Up Web Site, or Setup > SuiteCommerce Advanced > Set Up Web Site.
3. Click **Edit** next to the site you want to modify.
4. Check the **Take Web Site Offline for Maintenance** box.
5. Click **Save**.

Customers who visit your web site, see your custom maintenance page. When you are ready to put your web site back online, clear the **Take Web Site Offline for Maintenance** box, and then click **Save**.

 **Note:** If you check the **Take Web Site Offline for Maintenance** box and take your website offline, you cannot successfully rebuild the search index until you bring the website back online. If you attempt to rebuild the search index when your website is offline, the job status of the rebuild process remains in the Pending state until you cancel the job manually or until you bring the website back online. For more information on search index, see the help topic [Search Index](#).

Overview

i Applies to: SuiteCommerce Web Stores

You can optimize your website and decrease page-loading time by caching data and site assets. Caching enables content that is used more than once to reload quickly, increasing site performance and user satisfaction.

Caching occurs:

- On the Content Delivery Network (CDN)
- In the user's browser
- For product merchandising rules
- At the application level (for some previous implementations of SuiteCommerce Advanced)


See the following topics for more information about caching:

- [CDN Caching](#) – These topics explain how to optimize your website and decrease page-loading time by caching data and site assets on the CDN available with SuiteCommerce web stores. After you set up CDN cache, data and site assets that are reused across your web store are always cached.
- [Cache Invalidation](#) – Once your data is cached, it usually remains in the cache until it expires. These topics describe how to set up automatic or manual cache clearing, prior to data expiration, for when cached data changes at the source or when it should no longer be cached.
- [CDN for Secure Checkout Domains](#) – As CDN caching occurs on third-party servers, only content for which security is not of primary importance should be cached. This topic lists factors you should be aware of when using CDN caching for secure checkout domains.
- [Product Merchandising Rule Caching](#) – This topic describes how to reduce the default cache time before a new product merchandising rule becomes active.

CDN Caching

 **Applies to:** SuiteCommerce Web Stores

You can optimize your website and decrease page-loading time by caching data and site assets on the Content Delivery Network (CDN) available with SuiteCommerce web stores. After you set up CDN cache, data and site assets that are reused across your web store are always cached. Caching enables reused data and assets to be read quickly, increasing site performance.

 **Note:** CDN cache is not supported for Site Builder websites.


To learn more about CDN Caching, refer to the following sections:

- [Items Cached](#)
- [CDN Caching Times](#)
- [Limiting Cache](#)

Items Cached

The following items are cached by the CDN:

- All image files hosted in the file cabinet, identified by filename extensions such as, .gif, .jpg, .bmp, and .png, as well as static file system images that might be referenced from default site templates.
- Content in the file cabinet, marked Available Without Login.
- Static content (such as text) delivered from the file cabinet that does not contain custom Web Site tags.
- The response output from the Item Search API. For example, a request to the Item Search API that returns a list of items.
- The response output of executed SuiteScript hosted out of the file cabinet, if it sets response headers indicating the content is cacheable. This refers to SuiteScript associated with SSP applications.
- SSP files

 **Note:** Files stored in the file cabinet for sale as download items are excluded from the CDN cache.

CDN Caching Times

You can use the Cache Invalidation Request form to clear the CDN cache, but the browser cache remains untouched. For example, when you invalidate a URL, new customers see new content right away, but returning customers see old content until it expires in their browser cache.

If the CDN cache is not cleared, then CDN and browser cache expire at the same time.

The length of time that a certain type of content is cached also indicates the time it takes for returning customers to see updated content. Note the following:

Type of Content	Example	How long is it Cached?
Static Content	Images, CSS	7 days

Type of Content	Example	How long is it Cached?
Scripts	environment.ssp, getSlides.ss	2 hours
Content Delivery	page.ss	15 minutes

Limiting Cache

As you build your SuiteCommerce web store, it is important to see the changes you make in real time. After you set up CDN cache, you can use tools and strategies to ensure that you are looking at updated website content. For more information, read the following topics:


- [Bypass CDN Cache](#) – Use this strategy for setting up a subdomain for testing changes to your website.
- [Cache Invalidation Request](#) – Learn how to invalidate the cache for the whole domain or just the specified URLs.
- [Find URLs for Cache Invalidation](#) – Learn how to locate specific URLs for cache invalidation requests. This topic also includes guidelines for sending cache invalidation requests for different types of website updates, such as banners and stylesheets.

Enable CDN Caching

Before you enable CDN caching, you must have already set up a domain name for your website. For more information, see [Set Up Domains in NetSuite](#).

To enable the CDN cache:

1. Go to Setup > SuiteCommerce Advanced > Domains.
2. Click **Edit** next to the domain name that points to your main website.
3. On the Set Up Domains page, check the **CDN** box.
4. Click **Save** on the page to submit the change.

 **Note:** Alternatively, to go to the Set Up Domains page, you can navigate to Setup > Suite Commerce Advanced > Set Up Web Site, click **Edit** next to your SuiteCommerce site, click the **Domains** subtab, and then click **Edit** next to the desired domain.

Read the following topics about how to use CDN caching capabilities to assist you in web site development:

- After enabling CDN cache, data and site assets that are reused across your web site are cached for a certain duration of time. You can configure this time. For step-by-step instructions on configuring the CDN cache, read [Configure CDN Caching](#).
- For instructions on setting up a subdomain to bypass the CDN cache while testing your site, read [Bypass CDN Cache](#).

Configure CDN Caching

To configure the CDN cache, you must have already enabled the CDN cache. For more information, see [Enable CDN Caching](#).

To configure CDN caching:

1. Go to Setup > SuiteCommerce Advanced > Configuration.
2. Select the **Website** and **Domain**.
3. Click **Continue**.
4. Click the **Advanced** tab.
5. Click the **Cache** subtab.
6. Set the **Content Page CDN** cache duration for content pages.
7. Set the **Content Page TTL**.

This is the Time To Live (TTL) cache for content pages, specified in seconds. Value must be between 300 (5 minutes) and 7200 (2 hours).

8. Click **Save**.

To configure CDN caching (pre-Vinson):

These steps apply to pre-Vinson implementations of SuiteCommerce Advanced only.

1. Modify the SSP application.
2. Add the setCDNCacheable call within your SSP application and service files. For more information on using the setCDNCacheable method, see the help topic [setCDNCacheable\(type\)](#).

```
<% response.setCDNCacheable(response.CACHE_DURATION_MEDIUM) %>
```



Important: The Reference Shopping implementation has the setCDNCacheable call within the index.ssp file by default. After the domains are set up correctly as described above, CDN caching is enabled for Reference Shopping. When Reference Shopping is customized using external SSP applications, you can also set caching for content returned from those custom services.

After you enable and configure CDN caching, you can clear the cache by triggering a cache invalidation request. For more information, see [Cache Invalidation](#).

Bypass CDN Cache

During website development, you can bypass the CDN to view updated content on your site. To do this, you can set up **two domains** for your web store:

1. A domain that points to the CDN caching server (your main website domain)
2. A domain you use for testing that bypasses the CDN cache

You can visit each domain to verify that changes on your site appear as expected.

To set up a test domain for bypassing CDN cache:

1. Set up CDN for your main website domain. For more information, see [Configure CDN Caching](#).
2. Go Setup > SuiteCommerce Advanced > Domain.
3. Set up a new Domain. Enter a domain name that you can use for testing.
4. Do not check the CDN box as this test domain is intended to bypass the CDN server.

5. Do not select a **Primary Web Site URL** for either domain. This enables you to navigate directly to a website URL without redirecting to the “primary domain.”
6. Select the appropriate **HTML Hosting Root** and **Touch Point** for the subdomain.
7. Click **Save**.
8. Visit your domain provider's website to complete the task of setting up a CNAME record for your subdomain. For more information, see [Set Up a CNAME Record](#).

After you set up a domain that bypasses CDN cache, navigate to that domain for testing your site during web development. If you continue to see cached content on the domain pointing at the CDN server, use the domain pointed at NetSuite shopping servers to view updated content.


Cache Invalidation

 **Applies to:** SuiteCommerce Web Stores

Once your data is cached, it usually remains in the cache until it expires. If the data changes at the source or should no longer be cached, you might want to clear the cache prior to its usual expiration time. You can accomplish this by triggering a cache invalidation request through the Cache Invalidation Request form. For more information, see [Cache Invalidation Request](#).

In certain scenarios, automatic cache invalidation is triggered, which removes the need for manual intervention. For more information, see [Automatic Cache Invalidation](#).

You can view the status of all cache invalidation requests on the Cache Invalidation Status page. For more information, see [Cache Invalidation Status](#).

 **Note:** Browser cache is not affected by cache invalidation. For example, when you invalidate the cached content, new customers see the new content right away, but returning customers see old content until it expires in the browser cache.

Cache Invalidation Request

You can trigger a cache invalidation request for more than one domain at the same time. You can also choose to invalidate the cache for the whole domain or just the specified URLs.

Clearing Cache for the Whole Domain — As you develop your SuiteCommerce web store, you may have to redeploy SSP applications. To see changes to the content delivered from the file cabinet, script files, CMS, and Item Search API request output, use the Clear Cache for the Whole Domain option.

You can also choose this option if you have made multiple changes and you do not recall all URLs.


Clearing Cache for Specified URLs — Most of the content is cached in the SSP and SuiteScript method calls. To target specific SSP or SuiteScript files that contain the definition of the content you want to clear from the cache, use the Clear Cache for Specified URLs option.

For example, suppose you have a banner on your website that you plan to change frequently. Every time you wish to update this banner, submit a cache invalidation request using the URL of the SSP file (/page.ss) that includes the definition for this banner on your website.

To create a new cache invalidation request:


1. Go to Setup > SuiteCommerce Advanced > Cache Invalidation Requests > New.

The Cache Invalidation Request page is displayed.

 **Note:** You can also create a new Cache Invalidation Request by clicking **New Invalidation Request** on the Cache Invalidation Status page.


2. From the **Available Domains** list, select one or more domains to perform cache invalidation for. Note that must select at least one domain before you proceed.

Use the controls to move domains between the **Available Domains** box and the **Selected Domain** box.


 **Note:** SuiteCommerce Domains both with and without CDN are listed under Available Domains as long as the domains are in the Live state.

3. Choose the type of operation you wish to perform on the selected domains:

- **Clear cache for the whole domain(s)** — Clears cache for the whole domain.
- **Clear cache for the following path(s)** — Clears cache for the specified URLs. If you have selected the **Clear cache for the following path(s)** option, you must add at least one URL path before you click Submit.

 **Note:** For information about determining which URL you should use in cache invalidation requests, see [Find URLs for Cache Invalidation](#).

4. Click **Submit**.

 **Note:** For SuiteCommerce Domains without CDN, internal caches are cleared as a result of the cache invalidation request.


Cache Invalidation Status

Once you have created a Cache Invalidation Request, you can view its progress on the Cache Invalidation Status page. The Cache Invalidation Status page displays the status of all cache invalidation requests. If there are too many cache invalidation entries on the page, you can filter out the cache invalidation request that you are looking for by specifying the Submitted Date and Time, Request Type, Domain Name, User Name, and Status in the respective fields.

To view the status of a cache invalidation request:

1. Go to Setup > SuiteCommerce Advanced > Cache Invalidation Requests.

The Cache Invalidation Status page is displayed with a list of cache invalidation requests.

 **Note:** You can also view the status of a Cache Invalidation Request by clicking **Cache Invalidation Status** on the Cache Invalidation Request page.

2. (Optional) To filter the list of cache validation requests:
 1. Expand **Filters** on the Cache Invalidation Status page.
 2. Use the following fields to filter the list of cache invalidation requests:
 - **Date Submitted** — Filter the list of requests to include only ones in the specified date and time range.
 - **Request Type** — Filter the list of requests to include only Automatic requests, Manual requests, or both.
 - **Domain** — Filter the list of requests to include only the selected domains.
 - **User** — Filter the list of requests to include only the selected user.
 - **Status** — Filter the list of requests to include Processed requests, In Progress requests, Failed requests, or any combination of these statuses.
 3. (Optional) Click **Reset** to reset all the filter fields to the default values.
 4. Click **Filter Requests**.

Automatic Cache Invalidation

Automatic cache invalidation increases the accuracy of cached content by not serving obsolete data. It also removes the need for manual intervention every time you modify the content of the hosted files.

Automatic invalidation is triggered only in the following scenarios for files stored under File Cabinet > Web Site Hosting Files:

- When you modify the content of a hosted file
- When you overwrite a hosted file or media item

Note: Your CDN cache is automatically invalidated after every major release. This is applicable to all SuiteCommerce Advanced and SuiteCommerce Standard websites that have domains deployed on CDN.

In the context of hosted files, automatic invalidation is not triggered in the following scenarios:

- When you move, delete, or rename an existing hosted file or media item
- When you upload a new file

In situations where automatic cache invalidation is not triggered, you can manually invalidate the cache as described in [Cache Invalidation Request](#).

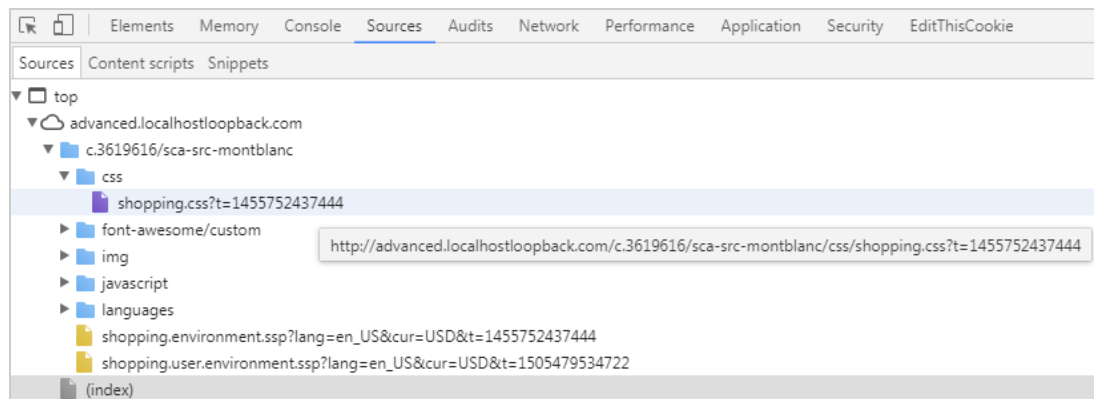
You can view the status of the triggered cache invalidation requests on the Cache Invalidation Status page. On the Cache Invalidation Status page, you can also filter out the cache invalidation requests triggered automatically by selecting **Automatic** in the Request Type field.

Find URLs for Cache Invalidation

The majority of data is cached in SSP and SuiteScript method calls, such as the method calls that exist in the `sc.environment.ssp` file. To effectively clear the CDN cache, you must clear the cache of URLs for the files that include those methods. Enter URLs that point to the appropriate SSP, CSS, or SuiteScript files in cache invalidation requests. You **cannot** clear CDN cache by using product URLs. Product URLs point to a specific item displayed on your website, such as <http://www.mywebstore.com/blue-hat>.

For more information about determining which URL you should use in cache invalidation requests, see [Cache Invalidation in Specific Scenarios](#).

The JavaScript console on your browser is an effective tool for finding website URLs based on the guidelines described below. After you activate the JavaScript console, click the Sources tab. Next, expand a folder to locate the URL for the file you want to use in cache invalidation requests.



Cache Invalidation in Specific Scenarios

The following guidelines describe how to invalidate caches in specific scenarios:

- Updating Stylesheets
- Changing Product Details
- Updating a Product Image
- Clearing Cached Site Management Tools Content

Note: The URLs provided here are examples only. Check the exact URL for your implementation before you submit an invalidation request. For information on how to find the correct URL, see [Find URLs for Cache Invalidation](#).

Updating Stylesheets

For optimum site performance, CSS stylesheets in SuiteCommerce websites are cached client-side for 7 days. You can submit a cache invalidation request to clear the CSS styles in CDN cache, but this does not clear client side browser cache of CSS styles. Therefore, when using cache invalidation to refresh CSS styles you must plan ahead to ensure both customers who have previously visited your site and new customers see the latest CSS changes.

For example, in two weeks you want to show a different color scheme on your website. Upload the new CSS defining the updated color scheme 7 days before you need it to go live. Then, submit a cache invalidation request for the URL of the current CSS file and URLs that reference the current CSS file.

To view updated CSS styles:

1. Upload the new CSS file to the file cabinet.
2. Use the Cache Invalidation by URL page to invalidate the cached CSS.
 - a. In the **Domain Name** field, enter the website domain. For example: **www.mywebstore.com**.
 - b. In the **URL** field, enter the URL for the CSS file. For example: **/c.7654321/shopflow-1-05-0/skins/standard/Styles-012345a6789.css**.
 - c. Click **Save**.

Changing Product Details

Code implementing Item Search API requests is cached for 5 minutes. CDN invalidation takes about 7 minutes. However, you can use the Cache Invalidation by URL page to force an update.

1. In the **Domain Name** field, enter the website domain. For example: **www.mywebstore.com**.
2. In the URL field, enter the Item Search API request URL. For example: **/api/items?id=199051&fieldset=search**.
3. Click **Save**.

Updating a Product Image

To update an image on your site, you do not need to use the Cache Invalidation by URL page. To view an updated image, you can check the **Generate URL Timestamp** box on the file record in the Web Site Hosting Files Folder.

The screenshot shows a 'File' management interface. At the top, there are buttons for 'Save', 'Cancel', and 'Reset', along with an 'Actions' dropdown. Below these, the file details are listed: 'FILE NAME' is '1 pixel.gif', 'FILE TYPE' is 'GIF Image', and 'FILE SIZE (BYTES)' is '814'. On the right side, there is a list of checkboxes: 'INACTIVE', 'AVAILABLE FOR SUITEBUNDLES', 'HIDE IN SUITEBUNDLE', 'AVAILABLE WITHOUT LOGIN', and 'GENERATE URL TIMESTAMP'. The 'GENERATE URL TIMESTAMP' checkbox is checked and highlighted with a red rectangle.

Checking this box forces the CDN to store a new version of your file. The file cabinet adds a URL time stamp based on the last date the file was modified. A new time stamp is generated each time the file changes. The URL time stamp indicates to the CDN and browser cache that the content has changed. For example, a new timestamp is generated each time you change an image used on a landing page or banner.

Best Practice: If you notice outdated images coming from the cache during website development, check the Generate URL Timestamp box on the file record. The search service reads the new URL, and the new version of the file is stored in the CDN cache.

Note: This feature only applies to URLs that are returned using code that implements the Item Search API, such as product images.

Clearing Cached Site Management Tools Content

Site Management Tools (SMT) content is cached just as any other content on your site. When new content is published SMT related caches are cleared. Cache clearing occurs for both immediate publication and scheduled publication.

The cache clearing task is specific to the account's site ID so that only the caches for the site where content was published is cleared. Caches for any other site owned by the account are not cleared.

The caches that are cleared include:

- Media Coherence Cache
- Ignite Response Cache
- Hosted Page Util Cache
- CDN Cache

CDN for Secure Checkout Domains


 **Applies to:** SuiteCommerce Web Stores

You can take advantage of the benefits provided by Content Delivery Networks (CDNs) for your secure custom-checkout domain. This topic lists items of which you should be aware prior to selecting to use a CDN at checkout.

CDN for Checkout Prerequisites

To enable CDN for checkout, your account must meet following conditions:

- **Environment** – CDN for checkout can be set up for your production account only. You cannot use it with a sandbox account.
- **Custom Checkout Domain** – You must provision at least one secure checkout domain using the Custom Checkout Domain add-on module. Custom domains are sometimes called **vanity** domains. You cannot use a CDN for `checkout.netsuite.com` or `youralias.securedcheckout.com` domains.
- **SSL Certificate upload**– You must upload your SSL certificate to NetSuite to your NetSuite **File Cabinet**. See [Purchase Domains and SSL Certificates](#) and [Upload your Certificate Files](#).

 **Note:** CDN for checkout is not available for Site Builder configurations.

CDN Caching for Private versus Public Content

The checkout process is a secure transaction that involves transmitting sensitive information. Caching keeps one or more copies of information on the network for a period of time. Because CDN caching occurs on third-party servers, caching should be limited to content for which security is not of primary importance.


Content that does not pose a security risk is considered public, while content that must be kept secure is considered private. Content designated as private should never be cached.

NetSuite uses private folders, public folders, and file extensions to determine which content is cached on CDN servers. To protect your shoppers, it is critical that you **do not maintain private checkout information in public folders**. Examples of checkout content that should be kept private include shipping addresses, billing addresses and credit card information.

Use this table as a guide to how folders and extensions determine which information may be cached by a CDN.


Path (folder) or extension	Cached on CDN
/private/ Private folder names must be lowercase. If a private folder name is all uppercase (PRIVATE), it is treated as a public folder and the contents will be cached. Uppercase folder names do not meet standard caching rules.	Never cached.
/public/	May be cached depending on header settings. If caching length not set,

	default TTL is 24 hours. Maximum is 7 days.
*.jpg, *.png, *.js, *.gif, *.css, *.html, *.ico, *.woff, *.shtml, *.txt, *.jpeg, *.ttf, *.woff2, *.json, *.jpeg, *.eot, *.pdf, *.xml, *.otf, *.cur, *.htm, *.tpl, *.swf	May be cached depending on header settings. If caching length not set, default TTL is 24 hours. Maximum is 7 days.
All other file extensions	Never cached. Considered to be private.

 **Warning:** Do not keep sensitive content in public folders because it may be cached on third-party servers.

CDN IP Address ranges

Instead of assigning a single IP address to your site, a CDN assigns a range of IP addresses. This range can and will change on a daily basis. NetSuite cannot predict or control which addresses the CDN will select. If you use your checkout domain for secure document storage, be aware that **you cannot whitelist an IP address** managed by a CDN.

 **Warning:** NetSuite cannot predict the IP addresses CDN providers will use to service *.netsuite.com requests.

For more information, see the help topic [Understanding NetSuite IP Addresses](#).

Product Merchandising Rule Caching

① Applies to: SuiteCommerce Web Stores

Product merchandising rules are cached to improve performance. The default cache time before a new merchandising rule becomes active is approximately two hours. If you want to reduce the cache time, you must make changes to the `DM_SS_Library.js` in the `Product Merchandising` bundle.

Caching for product merchandising rules is defined in two instances of the `TTL = attribute`. The default is set to `MAX_TTL`, which is the standard two-hour cache. To reduce the default time to approximately five minutes, change each instance to `MIN_TTL`.

Note that reducing caching times can impact performance.

```
function setTTL(paramValue){
    //ttl is not defined
    if(paramValue == null || paramValue == undefined || paramValue == ''){
        ttl = MAX_TTL;
        return ttl;
    }

    //ttl is defined but invalid
    var paramValueInt = parseInt(paramValue);
    if(isNaN(paramValueInt) || paramValueInt < MIN_TTL || paramValueInt > MAX_TTL){
        ttl = MAX_TTL;
        return ttl;
    }
    else{
        ttl = paramValueInt;
        return paramValueInt;
    }
}
```

Overview

i Applies to: SuiteCommerce Web Stores

Depending on your license agreement, you can run multiple websites on the same NetSuite account. Each website can have its own branding, look, and feel and can be used for different purposes such as separate websites for different brands, countries of operation, languages and so on.

If you have NetSuite OneWorld, your organization can use a single NetSuite account to manage records and transactions for multiple legal entities, or subsidiaries, conducting business across multiple tax jurisdictions, or nexuses, involving multiple currencies. You can then associate multiple subsidiaries to a single SuiteCommerce website, or you can have multiple websites for each subsidiary.

Multiple languages and currencies are also supported out of the box and websites can be customized to add translated components.

See the following topics for more information about microsites:

- [Multi-Site Development](#) – These topics explain how to set up multiple sites on the same NetSuite account.
- [SuiteCommerce and OneWorld](#) – These topics describe how to use NetSuite OneWorld with SuiteCommerce websites.
- [Localization](#) – These topics describe how to configure and manage translated components in SuiteCommerce web stores.

Multi-Site Development

 **Applies to:** SuiteCommerce Advanced

Depending on your license agreement, you can run multiple websites on the same NetSuite account. Each website can have its own branding, look, and feel. How you set this up depends on your implementation.

To set up your implementation for multi-site, be aware of the following tasks:


- [Configure NetSuite for Multi-site](#)
- [Organize Source Files for Multi-site](#)
- [Deploy a Specific Website](#)
- [Assign Customers to Websites](#)

Configure NetSuite for Multi-site

SuiteCommerce Advanced (SCA) supports using multiple website records, each with its own domain, or multiple domain configurations pointing to the same website record. When using multiple website records, you must configure item records to appear on each website. See the help topic [Item Configuration for Web Stores](#) for more information. When using multiple domains, each domain uses the same website record. All items configured for that website are available to both domains.

Organize Source Files for Multi-site

If you are implementing the Aconcagua release of SCA or later, best practice is to use themes and extensions to extend your website. Themes and extensions are available for installation as bundled SuiteApps or as custom themes and extensions deployed by in-house developers. After installing or deploying, the associated theme or extension can be activated for any domain associated with a website record. You must perform the activation for each site and domain you intend to use the theme or extension. No source code organization is necessary. See the help topic [Overview](#) for details.

 **Note:** For details on creating themes and extensions, see the help topics [Overview](#) and [Overview](#).

If you are implementing SuiteCommerce Advanced and not using themes and extensions, you must perform some set up tasks to allow multi-site development. You can also create custom modules that define different features for each website. You can also maintain different code bases for each website.

SCA lets you manage multiple websites using the same code base. This means that functionality that is common to each website uses the same core module code. Also, any customizations or extensions you create for a specific website are managed in the same way as other modules.

If you are customizing SCA and not using themes and extensions, see the help topic [Customize and Extend Core SuiteCommerce Advanced Modules](#) for information on creating custom modules.

For example if you were customizing the Header module for two different websites, you might have a directory structure as follows:

```
Modules
  suite_commerce
    <shared modules>
  custom_modules
```

```
Header_Site1
Header_Site2
```

In this example, the customized code is contained within the `custom_modules` directory. All source code shared among websites resides in the `Modules/suite_commerce` directory.

To determine which modules are included in a website, each website uses its own `distro.json` file. See the help topic [The `distro.json` and `ns.package.json` Files](#) for information about the `distro.json` file. The `json.distro` for a specific website is stored in a website-specific subdirectory within the `Sites` directory. The `Sites` directory is contained in the top-level directory of the SuiteCommerce Advanced source code. Thus, when managing multiple sites, you may have a directory structure containing directories similar to the following:

```
Modules
  suite_commerce
    <shared modules>
  custom_modules
    Header_Site1
    Header_Site2
Sites
  site1
    distro.json
  site2
    distro.json
```

You can compile and deploy the code for a specific website by running the gulp tasks from the site-specific subdirectory. For example, to compile and deploy Site 1, you would navigate to the `site1` directory and run the required gulp tasks. See the help topics [Deploy to NetSuite](#) and [SCA on a Local Server](#) for more information.

Deploy a Specific Website

 **Applies to:** SuiteCommerce Advanced

If you are implementing the Aconcagua release of SCA or later, best practice is to use themes and extensions to extend your website. You deploy your theme and extension to an account, where it is available for activation. See the help topic [Overview](#) for details on installing and activating themes and extensions. See the help topic [Overview](#) for information on using the theme and extension developer tools.

If you are implementing SuiteCommerce Advanced and not using themes and extensions deploy code to a specific website. You can deploy a website to the local server or to your NetSuite account.



Important: If you are implementing themes and extensions on an SCA site using the Aconcagua release or later, you do not deploy to a specific site. Any published themes and extensions are available as bundled SuiteApps. After these are installed, they can be activated for any domain on multiple sites.

To create and deploy custom websites (not using themes and extensions):

1. Install the developer tools and download the SuiteCommerce Advanced source code. See the help topic [Core SuiteCommerce Advanced Developer Tools](#) for more information.
2. Create any custom modules required for each website. [Create a Custom Module](#) for more information.

3. In the top-level directory of the SuiteCommerce Advanced source code, create the following:
 - A directory named Sites. This directory must have this name and be located at the same level as the Modules directory.
 - A subdirectory for each website.

4. Within each site-specific subdirectory, create a custom distro.json.

You can use the default distro.json file as a template. Your custom distro.json for each website points to the following:

- Shared module source code contains in `suite_commerce`.
- Customize modules defined in `custom_modules`.

5. Deploy your website.

To deploy a specific website, perform the following:

1. Navigate to the site-specific subdirectory for the website you want to deploy.
2. Deploy the website locally or to your NetSuite account.

You can deploy your website locally using gulp local or to your NetSuite account using gulp deploy. See the help topics [Deploy to NetSuite](#) and [SCA on a Local Server](#) for more information.

When deploying from a site-specific subdirectory, the gulp task creates the corresponding DeployDistribution and LocalDistribution directories within the site-specific directory. The compiled application files for the website are contained in these directories.

Assign Customers to Websites

If you run multiple websites, you may want to ensure that a customer who registers on a website only has login access on that particular site. To ensure that customers are automatically assigned to the website where they registered, you must first check the **Assign New Customers to this Web Site** box on the Web Site Setup page in NetSuite.

Note: Before you check the Assign New Customers to this Web Site box, customers are not assigned to any site. After you check the box, only new customers are assigned to the website. The checkbox does not affect existing customers.

Viewing the Source Website

You can use the **Source Web Site** field on each customer record to view the website where the customer has registered. Also, on the System Information subtab of the customer record, you can use the **Assigned Web Site** field, to change the customer's website assignment. If you want to change the website assignment for more than one customer, you can use a mass update operation.

Note: If you use custom customer record forms, you may not see the **Source Web Site** and **Assigned Web Site** fields on your custom form. You must customize the default customer form to enable these fields.

To enable automatic website assignment for customers:

1. Go to Setup > Site Builder/SuiteCommerce Advanced > Set Up Web Site..
2. If you use the Multiple Web Sites feature, click **Edit** next to a web site.

3. Click the **Shopping** subtab.
4. Check the box next to **Assign New Customers to this Web Site**.

Registration Page

☐ PASSWORD-PROTECT ENTIRE SITE

TYPE OF CUSTOMER REGISTRATION
optional

☐ DISPLAY COMPANY FIELD ON REGISTRATION PAGE

☐ MANDATORY

☐ CREATE CUSTOMERS AS COMPANIES

☒ ASSIGN NEW CUSTOMERS TO THIS SITE

5. Click **Save**.

After a customer is assigned to a website, login access is only allowed to that one site. You cannot assign a customer to more than one website. If you want to assign a customer to more than one website, then leave the Assigned Web Site field empty and allow the customer access to all of your websites.

To change a customer website assignment:

1. Go to Lists > Relationships > Customers.
2. Click **Edit** next to the customer record you want to modify.
3. Click the **System Information** subtab.
4. Make a selection in the **Assigned Web Site** list.

System Notes • Access • Active Workflows

☒ GIVE ACCESS

ASSIGNED WEB SITE
SuiteStyles

ROLE
Customer Center

To use Mass Update to change the customer website assignments in bulk:

1. Go to Lists > Search > Mass Update > Mass Updates.
2. Find **Customer Web Site Assignment** in the list, and then click **Update Customer Web Site Assignment**.
3. Make a selection in the **Assigned Web Site** list.
4. Select the criteria for the records you want to change.
5. Click **Save**.

After clicking Save, all customer records will be changed based on the criteria you selected.

For example, if you want to change all customers from Website A to Website B, then select Website B in the Assigned Web Site list (Step 3). In the Criteria list (Step 4), select Assigned Web Site for the filter, and then select Website A for the description.

Managing Customers and Website Assignments

After you start using customer website assignment capabilities, new customers are automatically restricted to having a customer account for the site on which they registered. Customers can register

on more than one website, but they will have more than one customer account. A new customer record is created with a different source web site when customers register on more than one website associated with your NetSuite account. You can set up duplicate website customer email detection to identify and manage duplicate customer records. For more information, read, [Detect Duplicate Customer Email](#).

Website Specific Customers and OneWorld

For companies that use OneWorld, a customer must be assigned to a website associated with a subsidiary that is online. For example, suppose you have two subsidiaries: Subsidiary A and Subsidiary B. Only Subsidiary B is marked online and associated with Website B. The customer linked to Subsidiary A cannot log into Website B when the Assigned Web Site is set to B or empty. Subsidiary A must be marked online for the customer to enable the customer to log in to a website.

SuiteCommerce and OneWorld

Applies to: SuiteCommerce Web Stores

With NetSuite OneWorld, an organization can use a single NetSuite account to manage records and transactions for multiple legal entities, or subsidiaries, conducting business across multiple tax jurisdictions, or nexuses, involving multiple currencies.

Although you can associate multiple subsidiaries to a SuiteCommerce website, only the subsidiary selected as the default shows up on the website.

To set a default subsidiary for a site, do the following:

1. Go to Setup > SuiteCommerce Advanced > Set Up Web Site.
2. At the bottom of the **Setup** subtab, in the **Default** column, select the radio button for the default subsidiary and click the **Save** button.

For example, when parent company Wolfe Electronics creates a website, the administrator selects the parent subsidiary as the default subsidiary on the Web Site Set Up page. Orders that are submitted on the website are posted to the parent subsidiary.

Move To Top		Move To Bottom	
ONLINE	DEFAULT	SUBSIDIARY	DISPLAY NAME
<input checked="" type="checkbox"/>	<input checked="" type="radio"/>	Wolfe Electronics US	Wolfe Electronics US
<input checked="" type="checkbox"/>	<input type="radio"/>	Wolfe Electronics West	Wolfe Electronics West

The site shows only the following types of information specifically associated with the default subsidiary with no option for the shopper to change the website subsidiary:

- Language options
- Currency options
- Minimum order amount
- Tax rates
- Shipping items
- Items

Visitors to the website also see items and shipping options associated with the default subsidiary. After registering or submitting an order, each customer created through the website is associated with the subsidiary set as the default.

Cross-Subsidiary Fulfillment

Applies to: SuiteCommerce Web Stores | Aconcagua

SuiteCommerce sites support Cross-Subsidiary Fulfillment for OneWorld accounts.

With Cross-Subsidiary Fulfillment, users can order an item from a subsidiary website regardless of inventory level at that subsidiary as long as there is sufficient inventory to fulfill the order at a related subsidiary. The inventory available to the customer equals the inventory level at the subsidiary with

the highest quantity of an item. It is not a combined total of inventory for the item at all subsidiary locations.

Examples:

Scenario One



- Subsidiary One has 15 blue widgets in inventory.
- Subsidiary Two has 25 blue widgets in inventory.

Customer on Subsidiary One website wants to order 10 blue widgets. The order is fulfilled from Subsidiary One because there is sufficient inventory at that location to fulfill the order.

Scenario Two



- Subsidiary One has 15 blue widgets in inventory.
- Subsidiary Two has 25 blue widgets in inventory.

Customer on Subsidiary One website wants to order 20 blue widgets. The order is fulfilled from Subsidiary Two.

Scenario Three



- Subsidiary One has 15 blue widgets in inventory.
- Subsidiary Two has 25 blue widgets in inventory.

Customer on Subsidiary One website wants to order 30 blue widgets. The order is not fulfilled because neither subsidiary has 30 blue widgets in inventory (even though the two subsidiaries have a combined total of 40 blue widgets in inventory).

Key Benefits:

- Potentially speeds up order fulfillment
- Sales and returns can be processed across more than one subsidiary. See the help topic [Processing Sales and Returns Using Intercompany Cross-Subsidiary Fulfillment](#).
- Reduces the entry and maintenance overhead associated with intercompany inventory transaction processing

To set up Intercompany Cross-Subsidiary Fulfillment for your SuiteCommerce site:

Note: If you have an existing SuiteCommerce website with items displaying correctly, skip step 1.

1. Set up items to display on your website. See the help topic [Item Setup for SuiteCommerce](#).
2. Enable the Intercompany Cross-Subsidiary Fulfillment feature. See the help topic [Enabling Intercompany Cross-Subsidiary Fulfillment](#).
3. Set up Global Inventory Relationship records (GIRs). See the help topic [Creating a Global Inventory Relationship Record](#).
4. Associate shipping items to reflect Global Inventory Relationships. See the help topic [Associating Subsidiaries with Shipping Items](#).

Localization

i Applies to: SuiteCommerce Web Stores

This section describes how to configure and manage translated components in SuiteCommerce Advanced (SCA). SCA supports multiple languages and currencies out of the box and can be customized to add translated components. Multiple language setup requires some frontend configuration.

Set Up NetSuite for Localized Content

To enable your website for localized content:

1. If setting up your site for multiple languages, verify that the Multi-Language feature is enabled:
 - a. In NetSuite, go to Setup > Company > Setup Tasks > Enable Features.
 - b. On the **Company** subtab, check the **Multi-Language** box.
 - c. Click **Save**.
2. Verify that all languages available on your site are defined in your Company General Preferences:
 - a. Go to Setup > Company > Preferences > General Preferences.
 - b. On the **Languages** subtab, select and add each language that you want to make available in your NetSuite account and on your website.
 - c. Click **Save**.

Only languages that have been enabled in your Company General Preferences are available. See the help topic [Configuring Multiple Languages](#) for more information.

3. Verify that all desired languages are enabled in the Web Site Setup page.
 - a. In NetSuite, go to Setup > SuiteCommerce Advanced > Set Up Web Site.
 - b. Click Edit for the desired website.
 - c. Navigate to the **Shopping** tab and **Languages** subtab.
 - d. Enable the checkbox in the **Online** column for each language you want enabled on the site.

Currency • Languages •		
ONLINE	DEFAULT	LANGUAGE
<input checked="" type="checkbox"/>	<input type="radio"/>	English (U.S.)
<input checked="" type="checkbox"/>	<input type="radio"/>	Français (France)
<input checked="" type="checkbox"/>	<input type="radio"/>	Español (España)
<input checked="" type="checkbox"/>	<input type="radio"/>	Português (Brasil)

- e. Verify that item names and descriptions have associated translation fields for each language.

When translation fields are populated in the item record, the corresponding translated fields display in the web store.

Note: Translation of custom fields on item records is not currently supported. Any data returned from custom fields for items in a Web Store will not have corresponding translated strings.

For detailed instructions, see the help topic [Multi-Language Names and Descriptions](#).

- f. Click **Save**.
4. Verify that all desired currencies are enabled in the Web Site Setup page.
 - a. In NetSuite, go to Setup > SuiteCommerce Advanced > Set Up Web Site.

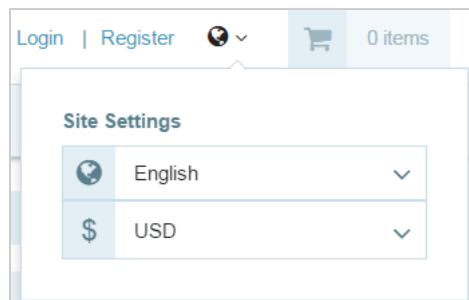
Note: Currencies must be set up in NetSuite to appear in this list. See the help topic [Working with Currencies](#) for information on adding currencies to NetSuite.

- b. Click **Edit** for the desired website.
- c. Navigate to the **Shopping** tab and **Currency** subtab.
- d. Enable the checkbox in the **Online** column for each currency you want enabled on the site.

Currency • Languages •		
<div>Move To Top</div> <div>Move To Bottom</div>		
ONLINE	DEFAULT	CURRENCY
<input checked="" type="checkbox"/>	<input checked="" type="radio"/>	USD
<input checked="" type="checkbox"/>	<input type="radio"/>	EUR
<input type="checkbox"/>	<input type="radio"/>	GBP
<input type="checkbox"/>	<input type="radio"/>	CAD

- e. Click **Save**.
5. If setting up your site for multiple languages and currencies, you must set up the `hosts` array. See [Configure Hosts for Sites with Multiple Languages](#).

After your website is set up and customized to support multiple languages, the location drop-down selector is available by default in the upper left region of your website.



Create Custom String Literals

Applies to: SuiteCommerce Web Stores

With the Vinson release of SuiteCommerce Advanced, NetSuite offers two ways to add new or customize existing string literals to change the default language values and corresponding translated values.

Method 1: Use the SuiteCommerce Configuration Record

 **Applies to:** SuiteCommerce Web Stores | Aconcagua | Kilimanjaro | Elbrus | Vinson

Use the SuiteCommerce Configuration record to specify new string literals that translate at runtime. NetSuite recommends this method when making small changes to translated content.

To configure properties in NetSuite:

1. Select the domain to configure at Setup > SuiteCommerce Advanced > Configuration.
2. In the SuiteCommerce Configuration record, navigate to the **Multi-Domain** tab and the **Translations** subtab.
3. Add custom strings in the available table as needed. See the help topic [Translations Subtab](#) for details.
4. Save the Configuration record.

Method 2: Extend the Application

 **Applies to:** SuiteCommerce Advanced | Kilimanjaro | Elbrus | Vinson | Mont Blanc | Denali

As an option, you can extend the application to change the string literals in the code and update the name-value pairs in the existing `SC.Translation` arrays. NetSuite recommends this method when making many changes to translated content.

You can also add new string literals that translate at runtime. When extending an application, ensure that new string literals you intend to localize are wrapped in the `_.translate()` function and that the corresponding language files are updated. If you have a small amount of customization to do, you can manually add the newly translated name-value pairs to the existing `SC.Translations` arrays for each language. However, if your web store includes many new string literals, NetSuite recommends scripting this process. Each time you compile your website code, all string literals wrapped in the `_.translate()` function should be processed to generate the appropriate name-value pairs in each language file.



Important: These language files must use **UTF-8** character encoding to ensure proper display of international characters.

Configure Hosts for Sites with Multiple Languages

 **Applies to:** SuiteCommerce Web Stores

The hosts array within the Configuration.js file provides language-specific and currency-specific site support for multiple languages and currencies. This functionality is disabled by default. To enable this feature, configure the host for each language and define specific properties.

Follow the correct configuration procedures below, depending on the version of SuiteCommerce you are implementing. For information on configurable properties, regardless of your implementation, see the help topic [Hosts Subtab](#).

Configure Hosts with the SuiteCommerce Configuration Record

i Applies to: SuiteCommerce Web Stores | Aconcagua | Kilimanjaro | Elbrus | Vinson

To configure the hosts array:

1. Select the domain to configure at Setup > SuiteCommerce Advanced > Configuration.
2. In the SuiteCommerce Configuration record, navigate to the **Multi-Domain** tab and the **Hosts** subtab.
3. Set feature properties as required.
4. Save the Configuration record.

Configure the hosts Array

i Applies to: Mont Blanc | Denali

To configure the hosts array (pre-Vinson):

1. Follow the steps presented in [Set Up NetSuite for Localized Content](#).
2. Set up a hosted domain for each additional language on your site. See the help topic [Overview](#) for details on how to create new hosted domains.
3. Setup the SspLibraries module for customization.
See the help topic [Customize and Extend Core SuiteCommerce Advanced Modules](#) for detailed information.
4. Open the Configuration.js file in the newly created module.
5. Within the `hosts` array, define a unique host for each location using the `title` property.
6. Define the `currencies` and `languages` arrays for each host using the properties defined in this section.

i Note: For detailed information on the hosts array in a pre-Vinson implementation, see the help topic [Hosts Array Parameters \(pre-Vinson\)](#).

For example, you want to set up a multiple-language site that contains three geographic locations: the United States, South America, and France. For each location, you want to define specific currencies and languages according to the following table:

Location	Currencies	Languages
United States	U.S. Dollar	English
South America	U.S. Dollar Argentinian Peso Uruguayan Peso	Spanish Portuguese English
France	Euro U.S. Dollar	French

You set up NetSuite to allow multiple languages and enable the applicable languages and currencies. You then set up hosted domains for each translated language.

The following code snippet shows what your customization might look like.

```
, hosts: [
```



```

{
  title:'United States'
,  currencies:[
    {
      title:'American Dollars'
    ,  code:'USD'
    }
  ]
,  languages:[
    {
      title:'English'
    ,  host:'en.mysite.com'
    ,  locale:'en_US'
    }
  ]
}
, {
  title:'South America'
,  currencies:[
    {
      title:'American Dollars'
    ,  code:'USD'
    }
    , {
      title:'Peso Argentino'
    ,  code:'ARS'
    }
    , {
      title:'Peso Uruguayo'
    ,  code:'UYU'
    }
  ]
,  languages:[
    {
      title:'Spanish'
    ,  host:'sa.mysite.dev'
    ,  locale:'es_ES'
    }
    , {
      title:'Portuguese'
    ,  host:'pt.sa.mysite.dev'
    ,  locale:'pt_BR'
    }
    , {
      title:'English'
    ,  host:'en.sa.mysite.dev'
    ,  locale:'en'
    }
  ]
},
{
  title:'French'
,  currencies:[
    {
      title:'Euro'

```

```

        , code:'EUR'
      }
    , {
      title:'American Dolars'
      , code:'USD'
    }
  ]
  , languages:[
    {
      title:'French'
      , host:'fr.mysite.com'
      , locale:'fr_FR'
    }
  ]
}
]

```

7. Deploy the module to your site.

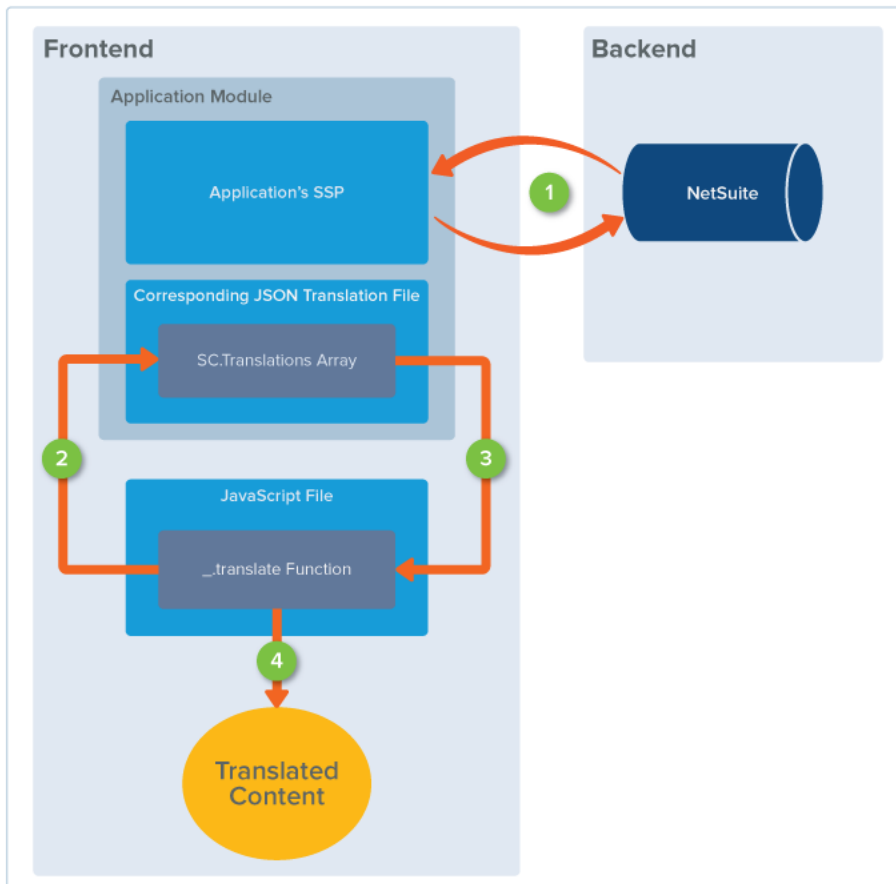
The Translation Process

Applies to: SuiteCommerce Advanced | Denali | Mont Blanc | Vinson | Elbrus | Kilimanjaro



Important: If you are localizing text using an extension for SuiteCommerce Standard or SuiteCommerce Advanced, see the help topic [Localize Text in an Extension](#).

When a specific language is set for a website, string literals wrapped in any `_.translate()` function are replaced with language-specific strings, which appear on the site. The following diagram depicts this process.



1. The application's SSP requests information from the backend to confirm that the requested language is properly set up in NetSuite.
2. All `_.translate()` functions are embedded within various JavaScript files and contain string literals to be translated. These functions match their string literals with name-value pairs defined in the `SC.Translations` array within JSON language files.

JSON language files are located in the Languages subfolder for each application module. For example, Modules > suitecommerce > ShoppingApplication@x.x.x > Languages.

Note: In the final Distribution folder, these files are located in the languages subfolder at the root level.

3. The name-value pairs within the `SC.Translations` array contain translated values for the active language. The `SC.Translations` array passes these values back to the `_.translate()` function.
4. The translated content appears on the site.

Important: If a literal wrapped in a `_.translate()` function does not have a corresponding name-value pair in the `SC.Translations` array for the active language, that literal is not replaced.

Note: In addition to the localization elements handled by the client, many elements are handled from the NetSuite backend that cannot be customized or extended. For example, NetSuite error messages, numbers for prices, dates, etc. are localized by the NetSuite backend code.

Pass String Literals

Important: If you are localizing text using an extension for SuiteCommerce Standard or SuiteCommerce Advanced, see the help topic [Localize Text in an Extension](#).

In the following code the string literal, **Country is required**, is translated to **Se requiere país** when the site's language is set to Spanish. This correlates to the JSON name-value pair defined in the `SC.Translations` array of the `es_ES.js` file within the application's Languages folder.

`_.translate()` function used in `Address.Model.js` file:

```
validation: {
  ...
  , country: { required: true, msg: _('Country is required').translate() }
  ...
}
```

Corresponding JSON used in `ShoppingApplication@x.x.x > Languages > es_ES.js`:

```
SC.Translations = {
  ...
  "Continue Shopping": "Continuar comprando",
  "Continue Shopping on our ${0}": "Continuar comprando en nuestro ${0}",
  "Country is required": "Se requiere país",
  "Credit Card Number is invalid": "Número de tarjeta de crédito no válido",
  "Credit Card type is not supported": "No se admite el tipo de tarjeta de crédito",
  ...
};
```

Note: Values from the `SC.Translations` arrays are passed off to a translation function. Therefore, there may not always be a direct relationship between the values set in the `SC.Translations` array for a specified language and the actual values displayed in your web store.

Within SuiteCommerce, many string literals are translated on the client side using the `_.translate()` function. When a specific language is set for a website, string literals wrapped in any `_.translate()` functions are replaced with language specific strings. These strings are retrieved from name-value pairs defined in the `SC.Translations` arrays within the JavaScript files located in the languages directory.

Language JavaScript files are located in the languages subfolder for each application module. For example, `Modules > suite_commerce > ShoppingApplication.x.x.x > Languages`. In the final Distribution folder, these files are located in the languages subfolder at the root level.

Note: If a literal wrapped in a `_.translate()` function does not have a corresponding name-value pair in the `SC.Translations` array for the active language, that literal is not replaced.

For example, in the following code the string literal `Country is required` is translated to `Se requiere país` when the site's active language is Spanish. This correlates to the JSON name-value pair defined in the `SC.Translations` array of the `es_ES.js` file within the `languages` folder.

`_.translate()` function used in `Address.Model.js` file:

```
validation: {
  ...
  , country: { required: true, msg: _('Country is required').translate() }
  ...
}
```

Corresponding JSON used in `ShoppingApplication@x.x.x > Languages > es_ES.js`:

```
SC.Translations = {
  ...
  "Continue Shopping": "Continuar comprando",
  "Continue Shopping on our ${0}": "Continuar comprando en nuestro ${0}",
  "Country is required": "Se requiere país",
  "Credit Card Number is invalid": "Número de tarjeta de crédito no válido",
  "Credit Card type is not supported": "No se admite el tipo de tarjeta de crédito",
  ...
};
```

Note: Values from the `SC.Translations` arrays are passed off to a translation function. Therefore, a direct relationship may not always exist between the values set in the `SC.Translations` array for a specified language and the actual values displayed in your web store.

Pass Parameters and HTML

Important: If you are localizing text using an extension for SuiteCommerce Standard or SuiteCommerce Advanced, see the help topic [Localize Text in an Extension](#).

In addition to passing string literals to the `_.translate()` function, you can also pass parameters and HTML segments.

Pass Parameters

In the following code, the variable values of `$(0)` and `$(1)` are passed 20 and 43 respectively.

```
_('Subtotal: ${0} of ${1}').translate(20,43)
```

Pass HTML

In the following code, an `HREF` segment is passed to the `translate` function so that the translated literal string is followed by the correct URL.

```
_('Continue Shopping on our $(0)').translate('<a href="/" data-touchpoint="home">' + _('Home Pa  
ge').translate() + '</a>')
```



Important: When customizing your applications with new `_.translate()` functions using parameters and HTML, ensure that your translation vendor does not translate the parameter or HTML code segments, only the literal strings.

Localize Content from the Content Delivery Service

i Applies to: SuiteCommerce Advanced

SuiteCommerce Advanced supports the locale tag, so you can deliver localized content generated from the Content Delivery bundle.

To localize content generated from the Content Delivery bundle:

1. Create a new Content Record for each localized version of the content to be displayed by the record.

For details on how to create a Content Delivery record see [Creating a Content Record](#).

2. Tag each record with the corresponding locale attribute.

For example, to specify a record as the American English version, use the locale tag **locale:EN_us**. The locale attribute must match the language/locale parameters set by the system when languages are enabled.

NAME * <input type="text" value="Ad Banner"/> <input type="checkbox"/> INACTIVE	SITE * <div> <div>MattSuite</div> </div>	CONTENT TAG <div> <div>- New -</div> <div>locale:EN_us</div> </div>
--	--	---

See [Using Content Delivery](#) for more detailed information on setting up your site for content delivery.

Overview

i Applies to: SuiteCommerce Web Stores

Integrating SuiteCommerce web stores with other NetSuite applications enables you to take advantage of the additional features they offer, such as:

- The ability for shoppers to access their NetSuite webstore account directly from your existing website without needing to log in again
- The ability to configure marketing automation features, such as post-purchase campaigns, on your webstore
- Easy access to NetSuite features for brick and mortar stores
- Easy creation of content pages and content areas on your website

See the following topics for more information about integration:

- [Single Sign-on Integration with External Websites](#) – Single Sign-on allows users to access your SuiteCommerce web store from an external website or access an external website from your SuiteCommerce web store without needing to log in to each site separately.
- [Bronto Integration](#) – Bronto is an Oracle company that provides an advanced marketing automation engine and solutions for shopping cart abandonment, post-purchase campaigns and so on.
- [SuiteCommerce InStore Integration](#) – SCIS is a web-based point-of-sale application available for the U.S. market. It is built on the same platform as SuiteCommerce web stores, and provides access to NetSuite records via an intuitive user interface designed for touch tablet screens.
- [Content Delivery Integration](#) – The Content Delivery SuiteApp combines customer forms, service files, and customer records to allow you to conveniently create and manage content pages and enhanced content areas for your website. Users who may not have a background in website scripting, such as marketing managers or site administrators, can integrate graphic elements, multimedia, and marketing content without writing code.

Single Sign-on Integration with External Websites

 **Applies to:** SuiteCommerce Web Stores

Inbound Single-Sign on (SSO) allows your web store users to log in to an external application and then move to a NetSuite web store without needing to log in again. Outbound SSO enables users to access an external application from your web store.

You can use either of the following methods to implement inbound SSO for your NetSuite web store:

- **SAML SSO** – This method uses authentication from a third-party identity provider.
- **Inbound SSO** – This method uses an encrypted token, and a dynamically constructed URL to redirect users from the external site to a NetSuite landing page..


Outbound Single Sign-on is available using the SuiteSignOn feature. For information, see [Outbound Single Sign-on \(SuiteSignOn\) Access from Your Web Store](#).


SAML Single Sign-on Access to Web Store

The SAML Single Sign-on (SSO) feature lets you set up SAML SSO web site access so that users who have logged in to an external application using SAML can click a link to go directly to a NetSuite web store. Users do not need to log in separately to the web store, because authentication from the same third-party identity provider (IdP) is used for login to both the external application and the web store. A user who accesses a web store using SAML SSO is directed to a landing page that you specify as part of SAML setup in NetSuite. SAML SSO access is supported for both SiteBuilder and SuiteCommerce Advanced web stores.

Any SAML 2.0-compliant application can serve as the IdP for SAML access to NetSuite web stores. You can use the same IdP for both web site access and NetSuite application access, or you can define different IdPs for each purpose. SAML Single Logout (SLO) functionality not supported for web stores. Specifically:

- IdP-initiated SLO is not supported.
- SP-initiated SLO is not supported.

 **Note:** The following potential solution is not part of the SAML 2.0 standard. If SP-initiated SLO is desired, and if your IdP supports this functionality, you could enter the Single Logout Service URL of your IdP in the Logout Landing Page field. There is no guarantee that this will work, as it depends on how your IdP implemented and supports the SAML SLO functionality.

 **Note:** For general information about SAML Single Sign-on with NetSuite, see the help topic [SAML Single Sign-on](#).

To set up SAML Single Sign-on for web store, first ensure that the SAML Single Sign-on feature is enabled at Setup > Company > Enable Features, on the SuiteCloud tab. You can then define other configuration parameters on the SAML subtab of the SSO subtab of the Web Site Setup page.

Important: The URL shown in the following screenshot in the NetSuite Service Provider Metadata field is obscured, because the URL varies depending on the data center where your account is hosted.

SAML Access Warning
By enabling the SAML Single Sign-on feature, you allow users to access and use your NetSuite account directly from a third party service that may not have the same authentication and security features as NetSuite. This feature also extends NetSuite administration of user access to the administrators of the identity management system. You need to ensure that NetSuite account use through SAML meets all of your security, regulatory, and other compliance obligations, including Payment Card Industry (PCI) Data Security Standards.

NetSuite Configuration

NETSUITE SERVICE PROVIDER METADATA
https://[redacted]/saml2/sp.xml

LOGOUT LANDING PAGE *

Landing Page After Login

Current Identity Provider

NOT SET

Set Up Identity Provider

SAMLV2 IDENTITY PROVIDER METADATA

☒ INDICATE IDP METADATA URL

☐ UPLOAD IDP METADATA FILE

Choose File No file chosen

- The **Landing Page after login** field is specific to SAML setup for web stores. By default, your site home page is the landing page for SAML users, but you can specify the URL for a different landing page in this field.
- Other fields on the SAML subtab of the SSO subtab of the Web Site Setup page are the same as those on the SAML Setup page for the NetSuite application available at Setup > Integration > SAML Single Sign-on. For details about these fields, see the help topic [Complete the SAML Setup Page](#).

If the Multiple Web Sites feature is enabled, you can set up SAML for different web stores by completing the SAML subtab of the Web Site Setup page for each web store.

Current Identity Provider

Entity ID [redacted]

[Delete IDP Configuration](#)

[Current Identity Provider Metadata](#)

You can use the same IdP for multiple web sites, as well as for the NetSuite application. You also have the option of defining different IdPs for each web site if needed. Click the links to view the **Current Identity Provider Metadata**, or to **Delete IDP Configuration**, if necessary.

Inbound Single Sign-on Access to Web Store

The inbound single sign-on feature can provide a seamless experience for your web store users, by allowing them to go directly from your user authenticating site to the web store, without having to log

in separately to NetSuite. For example, if you host your e-Commerce site externally and use NetSuite as the order processing service, you can route your customers directly from your site to the secure My Accounts area of NetSuite web store to check their order status.

Usually, a user would initiate inbound single sign-on access to the web store by clicking a link in your site. User credentials from your application are passed to NetSuite through an encrypted token, and are checked against NetSuite credentials to verify the user's identity. This check relies on a mapping between the two sets of credentials created by the web services maps operation.

After a user's NetSuite identity is verified, a dynamically constructed URL redirects the user from the external site to a web store landing page. By default, this page is your site home page, but you can specify a different page with a redirect URL parameter. If you use multiple sites, you can identify the site for this integration with another parameter.

NetSuite provides a downloadable kit that you can use to implement single sign-on. This kit includes tools to produce the OpenSSL keys used with the encrypted token, and to write integration code between your external site and the web store.

- For details about understanding and setting up this feature, see the help topic [Inbound Single Sign-on](#).
- To purchase this feature, contact your account manager.



Important: Inbound single sign-on is supported for custom checkout domains and multi-site implementations. You can define a custom checkout domain and/or a site ID for multi-site by setting URL parameters in single sign-on code. For details, see the help topic [Creating Single Sign-on Code Using SSOUrl Tables of Single Sign-on Redirect URL Parameters](#).

Inbound single sign-on also is supported for Web stores customized with SSP applications. For information about this type of customization, see [SSP Application Overview](#).

Outbound Single Sign-on (SuiteSignOn) Access from Your Web Store

The Outbound Single Sign-on feature, also called SuiteSignOn, enables user access to an external application from your web store. (See the help topic [SuiteSignOn Overview](#) for more information about this feature.)

The steps in setting up SuiteSignOn access to are to create a Suitelet, then create a single sign-on connection point. When the Suitelet is invoked (for example, from a menu) it will retrieve a response from the external application as configured on the Connection Points tab on the SuiteSignOn page.

For details about understanding and setting up the SuiteSignOn feature, see the help topic [Outbound Single Sign-on \(SuiteSignOn\)](#).

SuiteSignOn access is supported for SiteBuilder, SuiteCommerce Advanced, and reference implementation web stores.



Important: Only a Suitelet connection point is supported for SuiteSignOn access from your web store.

To view a code sample of a Suitelet used to reference the SuiteSignOn record, see the help topic [nlapiOutboundSSO\(id\)](#), Example 2.

Bronto Integration

 **Applies to:** SuiteCommerce Web Stores

Bronto is an Oracle company that provides an advanced marketing automation engine and solutions for shopping cart abandonment, post-purchase campaigns and so forth. Bronto can be easily integrated with your SuiteCommerce web store.

The following Bronto Applications are supported in SuiteCommerce:

- Cart Recovery
- Conversion Tracking
- Coupon Manager
- Pop-up Manager

To implement Bronto integration for your web store, you need to do the following:

- Determine whether your webstore uses a single secure domain for shopping and checkout. If not, you must ensure that the checkout domain is a subdomain of the shopping domain, for example:

Shopping: <http://www.mydomain.com>

Checkout: <https://secure.mydomain.com>


For information on how to acquire an SSL certificate and setup a custom checkout domain in NetSuite, see [Work with Custom Checkout Domains](#).

- Configure your site with your **Bronto** `accountId`. To do this:
 1. Go to Setup > SuiteCommerce Advanced > Configuration.
 2. Select the site that you want to configure from the **Select Website** list.
 3. Select the specific domain that you want to configure from the **Select Domain** list.
 4. Click **Continue**.
 5. Navigate to the Integrations > Bronto subtab and enter your Bronto accountId in the **Account ID** field.
 6. Click **Save**.

For more Bronto configuration settings, see the help topic [Bronto Subtab](#)

For details on updating the SuiteCommerce configuration files, see the help topic [Configure Properties](#).

With the `accountId` configured, the BrontoIntegration module causes the configuration to load after the SuiteCommerce Application has initialized. This configuration info file is cached in the Bronto CDN and uses the `accountId` to pull down the appropriate settings to enable and configure the various integrations setup in the Bronto Connector.

 **Note:** The BrontoIntegration module essentially just provides an interface to the Bronto CDN. Therefore, any updates or enhancements to Bronto applications are completely separate from SuiteCommerce releases and will be made available through the Bronto CDN. There is no additional customization needed in the SuiteCommerce source code to take advantage of Bronto enhancements.

For detailed information on configuring and managing your Bronto applications, refer to the Bronto documentation.

SuiteCommerce InStore Integration

Applies to: SuiteCommerce Web Stores | Mont Blanc | Vinson | Elbrus | Kilimanjaro | Aconcagua

SuiteCommerce InStore (SCIS) is a web-based point-of-sale application available for the U.S. market. It provides a touch-based user interface specifically designed for tablets.

SuiteCommerce purchases originate either online, on a SuiteCommerce web store, or via in-store (SCIS) sales. NetSuite manages these purchases using different types of Transaction records, depending on the origin. For example, an order placed online through your SCA site creates a Sales Order transaction record in NetSuite. Another purchase placed in your store using SCIS may require the Cash Sale or the Invoice record, depending on the transaction type.

When an SCA website is integrated with an SCIS platform, the SCIS transactions are incorporated into the SCA website order management and billing components. Shoppers can then:

- View all Online and In Store purchases in one list on their Purchase History page or filter results to display In Store purchases or Online purchases only. If viewing as one list, shoppers can differentiate between the two types of purchases using the **Origin** column.
- View and print a PDF of an individual purchase from the Purchase Details page.
- Determine if a purchase is associated with an existing sales quote.

Configure your Site for SCIS Integration

You must have the latest SuiteCommerce InStore (SCIS) and SuiteCommerce Standard or SuiteCommerce Advanced bundles installed and set up in your account before you can integrate them. For information, see the help topics [SCIS Installation](#) and [Install Your SuiteCommerce Application](#).

You must also set up NetSuite for SCIS Integration prior to configuration. For information, see the help topic [Enabling Features for SuiteCommerce InStore](#).

The correct configuration procedure depends on the version of SuiteCommerce you are implementing.

To configure your site for SCIS integration:

1. Select the domain to configure at Setup > SuiteCommerce Advanced > Configuration.
2. In the SuiteCommerce Configuration record, navigate to the **My Account** tab and the **SCIS Integration** subtab.
3. Configure the feature properties in this subtab as necessary.
4. Save the Configuration record.

To configure your site for SCIS integration (pre-Vinson):

1. Create a custom module that includes the backend Configuration object as a dependency. See the help topic [Configure Properties](#) for details.

Note: Do not edit the original Configuration.js source file directly. See the help topic [Customize and Extend Core SuiteCommerce Advanced Modules](#) for information and best practices on customizing JavaScript.

2. Redefine the `isSCISIntegrationEnabled` property in the custom module to true.
3. Save and deploy to your site.



Note: For information on configurable properties, regardless of your version of SCA, see [SCIS Integration Subtab](#).

Content Delivery Integration

 Applies to: SuiteCommerce Web Stores



Important: The Content Delivery Service is supported until the Denali R2 (revision 2) release of SuiteCommerce Advanced. Customers should use Site Management Tools instead of Content Delivery for Mont Blanc and greater.

The Content Delivery SuiteApp combines customer forms, service files, and customer records to allow you to conveniently create and manage content pages and enhanced content areas for your website. Users who may not have a background in website scripting, such as marketing managers or site administrators, can integrate graphic elements, multimedia, and marketing content without writing code.

Configuring Content Delivery Service

To configure the content delivery service:

1. Go to Setup > SuiteCommerce Advanced > Configuration.
2. Select the **Website** and **Domain**.
3. Click **Continue**.
4. Click the **Advanced** tab.
5. Click the **Cache** subtab.
6. Set the **Content Page CDN** cache duration for content pages.
7. Set the **Content Page TTL**.

This is the Time To Live (TTL) cache for content pages, specified in seconds. Value must be between 300 (5 minutes) and 7200 (2 hours).

8. Click **Save**.



Note: The Content Delivery bundle is supported for the ShopFlow, Checkout, and My Account applications. Content displayed in the Shopping application is delivered using this content delivery system. You can extend the SuiteCommerce Advanced applications by creating new content records for use with your own ecommerce solution. However, you can NOT extend the Content Delivery SuiteApp itself.

Understanding Content Delivery


There are two types of Content Delivery:


- **Landing Pages** : Allow you to create entire custom pages with custom URLs. You can specify as many different content areas within the new page and substitute content based on a set of pre-defined custom rules.

- **Enhanced Pages** : Allow you to supply custom content to existing pages. For example, for a page that returns item records based off of search criteria defined within the faceted navigation links you could add a descriptive content area to the top. This descriptive area could be further customized to display different content based off of attributes assigned to the enhanced page content area.

Content administrators create pages with containers to display content and define custom content tags to characterize page attributes such as language, delivery method, geography, or domain. After pages have been designed, new content records can be created with rules allowing users to decide how and when their content is displayed on the website. Users can create rules with various attributes to support marketing efforts or scheduled website changes.

Content is generated as JSON data and can be presented in your NetSuite-powered website, mobile applications, or 3rd party services, such as an externally hosted website or content marketing service.


 **Note:** Anchor Tags are not supported by Content Delivery.

 **Important:** Before defining Content Delivery pages, you must work with your developer to understand initial page design including content layout, content tag definitions, and css selectors used to call the content pages. Good initial planning will simplify the development and maintenance costs of your website and enable non-developers to define custom content when needed.

Using Content Delivery


Follow these steps to create new content areas for your website. To complete your new content areas, you also need to create **content records** and **Content Tags**. These records are defined on-the-fly from the Content Delivery page forms.

Content Delivery pages can be defined after you have coordinated with your developer to define the templates, content tags, and css selectors that will be used on your website. Refer to [Understanding Content Delivery](#).


 **Important:** Do not use the Content Delivery service to load external scripts into Landing Pages or Enhanced Pages. NetSuite recommends creating a custom module that includes the external script. See the help topic [Customize and Extend Core SuiteCommerce Advanced Modules](#) for details.

To add content delivery records:


1. Go to Setup > SuiteCommerce Advanced > Content Delivery.
2. Click on one of the following:
 - **New Landing Page**
 - **New Enhanced Page**
3. In the **Name** field, enter a name for this content page record.
This name does not display on your website. It is for record identification only. Each content record must have a unique name.
4. For **Landing Page Content** only, set the **URL** for this new page by entering a slash (/) followed by the desired URL. For example, /about_us.

 **Note:** The URL set here is amended to the selected site's URL to form the URL for this specific page. This URL must be unique to the site. If you are duplicating an existing URL you must deactivate the URLs you no longer want to display. You can NOT set a URL here when defining Enhanced Page Content.

5. If desired, enable the **Inactive** checkbox. An Inactive page can NOT be called from the website or previewed.
6. Select the website you want this page to display in. Select multiple sites by holding down the Ctrl key.

 **Note:** Content Delivery pages are only available in SuiteCommerce Advanced websites.

7. In the **Content Tag** field, select the tags that must be available to the page in order for this content to display.


 **Important:** Content Tags are used during customization and development of your website. Coordinate with your developer to obtain the list of available tags or to define a new set of tags. Content Tag names must match EXACTLY to the Content Tags as defined in the website assets by your developer.


Select existing tags or **New** to create a new tag. You can select multiple tags by holding down the Ctrl button.

Tags are NOT always required but may be used to further filter when the content area is displayed in your site. Content Tags reflect page attributes defined within the template files. When defining tags, enter the tag prefix (or group) followed by a colon and then the attribute.

For example you could have a group of device attributes or a group of language attributes.

device:mobile	locale:en_US
device:desktop	locale:es_ES
device:iphone	locale:fr_CA

 **Important:** In the case of **Landing Pages** with locales, at least one app Content Tag is required. You can select one tag or multiple tags. For Example, app:shopping, app:checkout, or app:myaccount. Your selection depends on which application will display the content.

 **Note:** Locale attributes are supported out of the box in Reference Implementations. When content is specified for a certain locale and a Web Site is configured for multiple languages, content is displayed for the language set by the current shopper. See [Localization](#) for more information.

8. For **Enhanced Pages**, add the following Content Tags: app:shopping, app:checkout and app:myaccount.

These tags are used to specify content for a specific application. You must add the tags even if you do not select them as the tags are also used internally when each application is interacting with Content Delivery pages.

9. For **Enhanced Pages** only, define the **Target URL**.

Pages at the target URL use the enhanced content records defined here to replace tags with the specified content. These are existing URLs in your website. You can target content to multiple URLs. You can also use wildcards to specify a group of pages.

For example, your site may have faceted navigation set up that allows users to filter on different types of shoes. When a specific URL based off of the selected filters is loaded, such as `/shoes/women/*`, then the enhanced content is displayed.

Note: This tab is not available when defining Landing Page Content. For Landing Page Content, a unique new URL must be provided as described in [Step 4](#).

10. Define Content Rules:

The screenshot shows the 'Content Rule' dialog box with the 'Advanced' tab selected. It features a 'Main Body' dropdown menu. Below it is a table with two columns: 'TARGET' and 'CONTENT'. At the bottom of the dialog are four buttons: 'Add' (with a checkmark icon), 'Cancel' (with an X icon), 'Insert' (with a plus icon), and 'Remove' (with a minus icon).

Content rules are used to define the content that will correspond to the current rule and where the content will display within the page. For Landing Pages, you may be defining content for each area within the page. For Enhanced pages you may have only a couple of areas being replaced with the content defined within these rules.

1. For **Landing Pages** only, select a content record to display in the **Main Body** of your page.

You can select an existing content record or create a new record by selecting New. See [Creating a Content Record](#) for more information.

Note: Selecting content in this field ensures that your page does not appear blank if no other rules are defined.

2. In the **Target** column, enter a container name.

Container names correspond to div areas defined in your Reference ShopFlow templates. The value entered here **MUST** match exactly to these div names. This information is provided by your website developer and corresponds to the area you want to display the selected content.

Note: Developers, you can refer to [Preparing Pages for Content Delivery](#) for more information.

3. Select the content to be displayed.

You can choose an existing content record or create a new record by selecting New. See [Creating a Content Record](#).

Note: In **Landing Pages**, the content record selected for the **Main Body** is NOT available for selection for other target areas.

11. Enter **Advanced** criteria.

1. Define page elements.

When this content is being created for a new Landing page, the page always loads with the information defined here. When created for Enhanced Page Content, the page is loaded only when the criteria for the enhanced page is met.

- **Page Title:** this title appears in the browser title bar.

- **Meta Keywords and Meta Description:** used to help search engines find your page.
- **Page Header:** This content replaces the existing header for the page.
- **Addition to <head> field:** enter any information you want to add to the <head> HTML element for this page.

For example, you could enter JavaScript that collects page hit information for marketing purposes.

2. Enter the **Name** of the template for this page.

Normally this can be left blank. However, if you have a reference shopping site with a custom template, indicate the template used for that site. This information is obtained from your website developer.

12. Click **Save**.

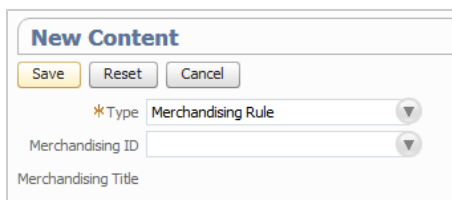
Creating a Content Record

Content records collect information about media and files you want to display. You then define delivery rules with the Content Delivery SuiteApp. The New Content Record is accessed when defining main body content for Landing Pages, or when defining Content Rules for either Landing Pages or Content Records as described in [Using Content Delivery](#).

When creating a new Content Record, you select one of two types, Merchandising Rule or HTML.

Using a Merchandising Rule

Only previously defined Merchandising Rules are available for selection. Simply select the desired rule and then click Save. For details on defining Product Merchandising rules, refer to [Product Merchandising](#).



Using HTML

You can use your own custom HTML elements to define the content displayed. HTML content can be defined for your record in one of two ways:

- **File:** Select this to use an existing HTML file. This file must first be uploaded into the file cabinet. Although you can store these files anywhere in your file cabinet, it is recommended that you keep them with other website files in the folder at Web Site Hosting Files > Live Hosting Files.
- **Text Editor:** Use the WYSIWYG editor to define the Content. Type your content in the Content field and using the built-in editor apply special formatting as desired.

Note: When the Type field is set to image, enter the url for the image. The url entered is an absolute path from your Live Hosting Files/site folder. This is normally the top images folder. For example /images/navbar.jpg.

You can also embed client-side JavaScript to add custom logic to your Content Delivery pages.

Note: If using jQuery methods in your logic you must use the `jQuery` prefix and not the `$` prefix. For example, use `jQuery(document).ready(function(){alert("Hello World");});` instead of `$(document).ready(function(){alert("Hello World");});`.

Important: Although you can add client-side JavaScript to Content Delivery records, you cannot reference **external** JavaScript libraries in Content Delivery records. External libraries must be referenced from the SSP application files or by using more comprehensive customization techniques.

Understanding Content Delivery Caching

When a page displays data from a content delivery record, that data is stored in the CDN cache for 2 hours by default. Changes to a content delivery record are not immediately reflected until this time has lapsed or the cache is cleared. For more information on CDN caching, see [CDN Caching](#).

You can manually clear the cache on the Web Site Setup page. Go to Setup > SuiteCommerce Advanced > Set Up Web Site > Edit. Then select Clear Cache from the More Actions drop-down menu.

SEO Considerations for Content Delivery

Content delivery enables you to manage important SEO related aspects of your website, including page titles and meta descriptions. See the following help topics for setting each of these SEO items with Content Delivery.

- [Setting Page Titles with Content Delivery](#)
- [Setting Meta Descriptions with Content Delivery](#)

Note: If you use Site Management Tools, you cannot use Content Delivery to set page titles or meta tags. Instead, those are set on Site Management Tools landing and enhanced pages. See the help topic [Landing and Enhanced Pages](#).

Setting Page Titles with Content Delivery

Page title are an important aspect of your site's SEO. The title is the main label for the page. Not only is the title displayed in the web browser's title bar or browser tab name, but the title it is also listed on search engine search results pages. See the help topic [SEO and Page Titles](#)

If you use Content Delivery, use the following procedures to set page titles for the pages on your site.

- [Home Page Title](#)
- [Item Detail Page Title](#)
- [Facet Search Page Title](#)
- [Landing Page Title](#)

Home Page Title

The title for the home page of your site can be set with a SuiteCommerce Advanced custom module. See the help topic [Home Page Title](#).

Item Detail Page Title

The title for an item detail page is set on the Item Record in NetSuite. See the help topic [Item Detail Page Titles](#).


Facet Search Page Title

You can use Content Delivery enhanced pages to set SEO friendly page titles for your search pages.

To set facet search page titles with content delivery enhanced pages:

1. Go to Setup > SuiteCommerce Advanced > Content Delivery.
2. Click **New Enhanced Page**.
3. In the **Name** field, enter a name for this enhanced page.
4. In the **Site** list select the website for which you are creating the enhanced page.
5. On the Target URL subtab, in the **Target URL** field, enter the URL for which you want to specify a page title.
6. Click **Add**.
7. Select the Content Rule subtab.
8. In the **Target** field, enter **#empty**.
9. In the **Content** dropdown list select **New**. Create a new blank HTML page.

10. On the Advanced subtab, enter the title for the page in the **Page Title** field.

 **Note:** On this page you can also set other important SEO elements for the page, such as the `meta description`.

Landing Page Title

You can use Content Delivery to set SEO friendly titles for landing pages. For more information on Content Delivery landing pages, see [Using Content Delivery](#).

To set landing page titles:

1. Go to Setup > SuiteCommerce Advanced > Content Delivery.
2. Click the **edit** link for the landing page.
3. On the Advanced subtab, enter the title in the **Page Title** field.
4. Click **Save**.

Setting Meta Descriptions with Content Delivery


The `meta description` for the pages on your site are an important SEO consideration because they are used to describe the page to search engines. This description is also commonly used as the summary of the page on search engine search results pages. See the help topic [SEO and Meta Descriptions](#). You can use Content Delivery to set the `meta description` for the home page, facet pages, and Content Delivery landing and enhanced pages.

Meta Description for the Home Page


You can use a Content Delivery enhanced page to add the `meta description` to the home page of your site. This description should be as succinct and informative as possible to entice visitors to visit your site.

To set the meta description for the home page:

1. Go to Setup > SuiteCommerce Advanced > Content Delivery.
2. Click **New Enhanced Page**.
3. In the **Name** field, enter a name for this enhanced page, for example **home page**.
4. In the **Site** list, click the site for this enhanced page.
5. In the **Target URL** field on the Target URL subtab, enter `/`. This denotes the home page of your site.
6. On the Content Rule subtab, enter **#empty** in the **Target** field.
7. Select **New** from the **Content** dropdown list. This displays the new content window. Create a new blank HTML page.

 **Note:** If you already have a blank HTML page, you can use it instead of creating a new one.

8. On the Advanced subtab, in the **Meta Description** field, enter the text to use as the `meta description` for the page.


 **Note:** You can also set other attributes, such as page title and page header on this tab.

Meta Description for Facet Pages


You can use Content Delivery to create the `meta description` for a facet search page. For example, for a facet filter of men's shoes, your description might be **Shop the best deals on men's dress shoes, loafers, sneakers, boots, and more at YourSite.com.**

To set the meta description for a facet page:

1. Go to Setup > SuiteCommerce Advanced > Content Delivery.
2. Click **New Enhanced Page**.
3. In the **Name** field, enter a name for this enhanced page, for example **mensshoes**.
4. In the **Site** list, click the site for this enhanced page.
5. In the **Target URL** field on the Target URL subtab, enter the URL for the facet listing. For example, if the URL for Men's Shoes is `/mens/shoes`, then enter `/mens/shoes`.
6. On the Content Rule subtab, enter **#empty** in the **Target** field.
7. Select New from the **Content** dropdown list. This displays the new content window. Create a new blank HTML page.

 **Note:** If you already have a blank HTML page, you can use it instead of creating a new one.

8. On the Advanced Subtab, in the **Meta Description** field, enter the text to use as the `meta description` for this facet page.

 **Note:** You can also set other attributes, such as page title and page header on this tab.

Meta Descriptions for Content Delivery Landing Pages

The `meta description` for a Content Delivery landing page is specified on the Content Delivery landing page record. You enter the `meta description` in the **Meta Description** field on the **Advanced** subtab. For more information on creating landing pages, see [Using Content Delivery](#).

Preparing Pages for Content Delivery

This section describes how to prepare Reference Implementation pages for content delivery. Preparing pages for Content Delivery should be done by your web application developers as it involves editing code within the related page.

Configuring the Reference Implementation

To display Content Delivery pages from your site pages, you must first configure the SuiteCommerce Advanced SSP to include the content delivery scripts.

When multiple Reference Applications are being used for your web store you can specify which applications use a Content Delivery record by using the "app" tag with the name of the implementation

as follows: Checkout, MyAccount, or Shopping. For example, using the tag `app:Shopping` causes the content to display in the Shopping Application only. If no app tag is used, the content is available for all Reference Applications.

You can also include content generated from the Content Delivery service inside Backbone Modals. To use Content Delivery content inside a modal, the target ID defined in the Content Delivery record must be referenced by adding the prefixViewId "in-modal-". You can also specify whether that content should display only when inside a modal, when inside or outside of a modal, or only when outside of a modal.

To include the Content Delivery scripts:

1. Go to Setup > SuiteCommerce Advanced > SSP Applications.
2. Click **Edit** next to the SuiteCommerce Advanced SSP application used for the web site where content delivery is used. For example, SuiteCommerce Advanced — Dev [version].
3. In the **Libraries** subtab under **Scripts**, click **Add**.
4. Navigate to the path of the `content_delivery_service_library_v2.js` file in the Content Delivery reference installation.



Important: If you are using a custom ssp application, you will need to define this library for your custom ssp application.

5. Click **Save**.

Overview

 **Applies to:** SuiteCommerce Web Stores

SuiteCommerce site security features enable you to monitor changes made to your website and protect it from unauthorized access. You are able to:

- Restrict customer access to specific website features such as Order History
- Restrict access to your entire website
- Restrict access to price information
- Track changes that have been made to your website records

See the following topics for more information about site security:

- [Feature Access Permissions](#) – To restrict access to specific features, you need to modify the Customer Center role permissions for each user.
- [Website Restriction](#) – You can restrict access to either your entire website or to pricing information.
- [Website System Notes](#) – System notes track changes that have been made to your website records.

Feature Access Permissions

① Applies to: SuiteCommerce Web Stores

SuiteCommerce uses the Customer Center permissions to enable some features and generate menu items for the My Account application. These permissions are configured in the NetSuite account containing the website record for your site. See the help topic [Custom Centers](#) for more information.

My Account Menu Permissions

The My Account application contains a menu that provides links to different features within the application. This menu is available in the following locations:

- The sidebar menu.
- The application header.

SuiteCommerce generates each of these menus dynamically based on feature areas the user has permissions to access. By default, SuiteCommerce uses the NetSuite roles and permissions of the currently-logged-in user to determine which menu actions are displayed. For example, if a user does not have permission to view the Order History in NetSuite, the Order History & Returns link is not displayed in the My Account menu for that user.

However, the mechanism for generating the menu items based on permissions is different for each menu. Permissions attributes are set for each view as needed and then in the template files those permissions attributes are used to determine which links to display.

Header Menu

The Header application module displays the application header. The `header_menu_myaccount.tpl` template defines each menu and submenu item in the HTML. To determine which menu items the user has permissions to view, the HTML tests the permissions based on the values of the `data-permissions` attribute.

For example, in the following code snippet from `header_menu_myaccount.tpl`, the `div` is displayed only when the permissions for the currently logged in user matches both `data-permissions` attributes listed.

```
<a class="header-menu-myaccount-anchor-level3" href="#" data-touchpoint="customercenter" data-hashtag="#returns" data-permissions="transactions.tranFind.1,transactions.tranRtnAuth.1" name="returns">
    {{translate 'Returns'}}
</a>
```

Account Overview Menu

The Account Overview menu is generated automatically when the My Account application is loaded. Each application module that defines a feature with a corresponding menu item returns an object called `MenuItems`. This object is generally defined within the entry point module. For example, the following code snippet from the `Quote.js` file defines the menu item that appears in the Account Overview menu:

```

MenuItems: {
  parent: 'orders'
  , id: 'quotes'
  , name: _('Quotes').translate()
  , url: 'quotes'
  , index: 5
  , permission: 'transactions.tranFind.1,transactions.tranEstimate.1'
}

```

After all modules are loaded into the application, the MyAccount.on method of the SC.MyAccount module loads each menu item into an array.

Forbidden Access

Although the My Account application only enables features that a user has permission to access, it is possible they may attempt to access a page they do not have permission to view. The ErrorManagement.ForbiddenError.View module of the ErrorManagement application module displays an error message if a user does not have permission to access a feature. By default, the template for this view displays the following error message:

Sorry! You have no permission to view this page. Please contact the website administrator, click [here to continue](#).

You can customize this template as needed for your web store.

Customer Center Custom Permissions


When customizing an existing module or creating a new one, ensure that the appropriate permissions are granted to the customer center role. See the help topic [Customizing the Customer Center Role](#) for more information.

The following table lists the SuiteCommerce features that require custom permissions. This table lists the default minimum permissions required for each item in the Customer Center. When customizing or creating a feature, you may need to test the combination of permissions and levels to ensure the feature works correctly.

Feature	Subtab	Permission	Level
Orders menu	Transactions	Find transactions Sales orders	View View
Returns menu item	Transactions	Find transactions Return authorizations	View View
Quotes menu item	Transactions	Find transactions Estimates	View View
Invoices menu item	Transactions	Find transactions Invoice	View View
Transaction history menu item	Transactions	Find transactions Additionally, you must grant permissions for at least one of the following: Invoice Credit Memo Customer Deposit Deposit Application	View

		The combination of permissions granted determines the items of the transaction type filter that is displayed.	
Print statement menu item	Transactions	Generate Statements	Create
Cases menu	Lists	Cases	Create
Continue to payment button (account balance)	Transactions	Customer Payment Invoice	Create View
Print statement button (account balance)	Transactions	Generate Statements	Create
Request return button (invoice details)	Transactions	Return Authorization	Create
Make a payment (invoice details)	Transactions	Customer Payment Invoice	Create View
Make a payment (invoice list)	Transactions	Customer Payment Invoice	Create View
Request a return (order history)	Transactions	Return Authorizations	Create
Recent orders (overview)	Transactions	Find Transactions Sales Order	View View

Website Restriction

 **Applies to:** SuiteCommerce Advanced | Mont Blanc | Vinson | Elbrus | Kilimanjaro | SuiteCommerce Web Stores

This feature is available in the **Mont Blanc** release of SuiteCommerce Advanced and later.

SuiteCommerce Advanced lets you restrict access to your website in two ways:

- [Restrict Access to Your Entire Site](#)
- [Restrict Access to Pricing Information](#)

Both options are disabled by default and require some minor configuration to function.



Warning: The Search API is public and contains all the information exposed by the fieldset being used. Enabling password protection restricts users from accessing some or all of your site. However, a user who knows a URL that calls the Search API could access some of this information.

Restrict Access to Your Entire Site

You can restrict your entire site to logged in users only. If this feature is enabled, visitors must register and log in to see the contents of your site. The site redirects unauthenticated visitors to a Login page with a link to a Registration form, where they must register on your site to gain access.

Follow the correct set up procedures, depending on the version of SuiteCommerce Advanced you are implementing.

To set up NetSuite:

1. In NetSuite, go to Setup > SuiteCommerce Advanced > Web Site Setup.
2. Go to the **Shopping** tab.
3. Check the **Password-protect Entire Site** field.
4. Set the **Customer Registration Is** field to any selection of your choice other than **disabled**.



Important: If this field is set to **Disabled**, this feature will not work.

To set up NetSuite (pre-Vinson):

1. In NetSuite, go to Setup > SuiteCommerce Advanced > Web Site Setup.
2. Go to the **Shopping** tab.
3. Set the **Customer Registration Is** field to any selection of your choice other than **disabled**.



Important: If this field is set to **Disabled**, this feature will not work.

4. Create a custom module that includes the backend Configuration object as a dependency. See the help topic [Configure Properties](#) for details.



Note: Do not edit the original Configuration.js source file directly. See the help topic [Customize and Extend Core SuiteCommerce Advanced Modules](#) for information and best practices on customizing JavaScript.

5. Redefine the `passwordProtectedSite` property in the custom module to true.

6. Save and deploy to your site.

Known Limitations


Before restricting access to your SuiteCommerce Advanced website, be aware the following information:

- When a user clicks on a item's Quick View link after their session has ended, SuiteCommerce Advanced redirects the user to the Checkout page instead of a Login page. The user cannot access the site, however, without logging in.
- If a user changes the language after logging into a password-protected site, the application recognizes the new domain and redirects the user to log in again.

Restrict Access to Pricing Information

You can restrict pricing information to registered users only. If this feature is enabled, visitors can view your site, but they must log in to view prices and purchase products or services. This feature lets you hide prices from competitors or wholesalers and encourages users to register on your site.

With this feature enabled and a user is not logged in, SuiteCommerce Advanced replaces prices with a log in message and link. The user can follow this link to access a Login/Registration page, where they must register on your site to see prices or make purchases.

 **Note:** Restricting access to your entire site overrides this feature.

The correct set up procedure depends on the version of SuiteCommerce Advanced you are implementing.

To restrict access to pricing information:

1. In NetSuite, go to Setup > SuiteCommerce Advanced > Web Site Setup.
2. Go to the **Shopping** tab.
3. Check the **Require Login for Pricing** field.

To restrict access to pricing information (pre-Vinson):

1. Create a custom module that includes the backend Configuration object as a dependency. See the help topic [Configure Properties](#) for details.

 **Note:** Do not edit the original Configuration.js source file directly. See the help topic [Customize and Extend Core SuiteCommerce Advanced Modules](#) for information and best practices on customizing JavaScript.

2. Redefine the `loginToSeePrices` property in the custom module to true.
3. Save and deploy to your site.

Website System Notes

You can use system notes to track important changes to website implementations. System notes are visible to account administrators, as well as any user with access to the Web Site Setup page.

To view system notes:

1. Go to Setup > SuiteCommerce Advanced > Set Up Web Site.
2. Click **Edit** next to the website.
3. Click the **System Notes** subtab.

System notes on the website record are also exposed to Saved Search. To create a saved search, click System Note in the New Saved Search list, and then filter the search for the website record type.

To create a saved search of changes to the Web Site record:

1. Go to Lists > Saved Search > New.
2. Click **System Notes**.
3. On the **Criteria** subtab in the list of **Filters**, select **Record Type**, and then select **Website**.
4. Click **Save & Run**.

The search results display all the changes made to your websites including the date when the change was made and by whom.

 **Note:** Not all changes made to the Web Site Setup page are logged in system notes.

Overview

You can create SuiteScript Server Page (SSP) applications to add a custom application to your web store. SSP applications are made up of a front-end page that executes server side SuiteScript.

See the following topics for more information about SSP applications:

- [SSP Application Overview](#) – This topic gives you a broader understanding of SSP Applications
- [Components of SSP Applications](#) – This topic discusses the different components of SSP Applications.
- [Working with SSP Applications](#) – The topics here teach you how to create an SSP Application.
- [Sample SSP Application Code](#) – This topic provides you several different code samples for SSP Applications.
- [Integration with Third Party Checkout Providers](#) – you can integrate your SSP Application with third party providers such as Paypal express. This topic teaches how to create the request handlers and call them from your SSP Application.
- [Debugging an SSP Application](#) – When you create your SSP application you can include debugging. Learn how to write debugging details to SSP Application record log.
- [Bundling an SSP Application](#) – After you finish creating an SSP Application, you can bundle it for installation in other NetSuite accounts. Consult this topic to learn how.


SSP Application Overview

SuiteScript Server Pages (SSP) is the custom web application platform used to build applications for SuiteCommerce Advanced. SSP applications are an iteration on the Suitelet, allowing customers to build frontend pages that execute server side SuiteScript to generate dynamic content. The technology emulates JSP, but uses SuiteScript as the underlying mechanism. Also, SSP applications can be bundled and distributed in full as SuiteApps, allowing for the easy sharing and sale of custom applications.

SSP applications consist of .ssp files and a library of JavaScript files. The SuiteScript Server Pages (.ssp files) make AJAX calls to SuiteScript files and support a clear delineation between frontend (website display) and backend (business) logic. Supported touch points, such as View Homepage, Log In, and Log out are defined on the SSP application record. All files in an SSP application are grouped by author (application publisher), and stored together in the file cabinet in the Web Site Hosting files folder.

The SuiteScript Server Pages feature is available for both Site Builder and SuiteCommerce Advanced websites. This feature supports the packaging of SuiteCommerce website assets as SSP applications. You can create SSP applications to customize your e-commerce website, and then link your site to SSP applications by selecting touch points to specify entry page URLs.

For more information, see [Working with SSP Applications](#).

 **Note:** Inbound single sign-on access is supported for web stores customized with SSP applications. See [Inbound Single Sign-on Access to Web Store](#).

SSP Application Structure

Each .ssp file represents a page that can replace a supported web store touchpoint. An .ssp file includes presentation logic for the touchpoint. These .ssp files also allow you to add HTML and generate dynamic content to be rendered in the browser.

Backend services are included in .ss files, typically one .ss file per service. Ajax calls from an .ssp file provide access to .ss file services. Note that in addition to the SuiteScript API, both .ssp and .ss files can use the Commerce API. For more information, see [File Types for SSP Applications](#).

SSP Applications and SuiteCommerce Advanced

SuiteCommerce Advanced is an e-commerce website and checkout solution delivered in a set of SSP applications, available as a SuiteApp that you can install in your account. Files associated with SSP applications can be modified, tested and debugged in your NetSuite account. This architecture enables you to manage multiple SSP applications that customize a single web store. SSP applications ensure that all assets are linked together. This architecture also supports access using named URLs.

Components of SSP Applications

To customize and create SSP applications successfully, it is important to understand the different components involved. Note the following:

- [SuiteScript Server Pages Feature](#)
- [SSP Application Record](#)
- [SSP Application Precedence](#)
- [File Types for SSP Applications](#)
- [Commerce API](#)

SuiteScript Server Pages Feature

To create, view, or customize SSP applications, you must turn on the SuiteScript Server Pages feature. For more information, see [Enabling Required Features for Using SSP Applications](#).



Important: You must have the SuiteScript permission to access SSP Application records. To set up permissions, go to Setup > Users/Roles > Manage Roles.

SSP Application Record

The SSP Application record provides a single NetSuite object that stores details about a website customization. This record facilitates grouping of assets, debugging of script files, and packaging of the customization for use in other NetSuite accounts. On this record, you can create and view the following properties:

- **Application Folder** – the file cabinet folder where customization files are located. For more information, see [Creating an SSP Application Folder](#).
- **URL Root** – the root used in web store links to SSP application assets. For more information, see [Creating an SSP Application Record](#).
- **Supported Touch Points** – the pages in the web store that should be replaced with pages from the SSP application. For more information, see [Selecting Supported Touch Points](#).

SSP Application Precedence

When SSP applications share a URL and hosting root, a system of precedence determines which SSP application handles a particular request. For more information, see [Changing SSP Application Precedence](#).

File Types for SSP Applications

Web store customizations implemented through SSP applications support the following file types:

- **.ssp files:** These files, known as **SuiteScript Server Pages**, are frontend pages that can be written in HTML and/or server-side SuiteScript. They are similar to .jsp files, use a similar tag convention, and support Includes so that code can be pulled from associated libraries. You may invoke suitelets and RESTlets from .ssp files. Also, you can use .ssp files to develop pages that use JQuery.

- **.ss files:** These files are request handlers written in SuiteScript, basically a specialized type of suitelets. They can be frontend non-HTML pages, such as AJAX handlers, or they can be backend pages that take an HTTP request and perform an action, such as writing HTML, XML, JSON or JavaScript to the output stream or issuing a redirect.

Both .ssp files and .ss files run against the server. These files support the full range of existing SuiteScript APIs that run on the server, as well as the Commerce API designed to support web store customizations.

For basic examples of .ssp and .ss file content, see [Sample SSP Application Code](#).

Commerce API

The API, supported by both Site Builder and SuiteCommerce Advanced, includes all of the web store-specific objects and related methods that you need to fully customize your web store on the SuiteCloud platform.

For details about this API, see the help topic [Commerce API](#).

Working with SSP Applications

Use an SSP application to customize an e-commerce website. First, you must enable the required features. Next, assemble all the assets you need to build the SSP application, including SuiteScript to handle requests for NetSuite data, and .ssp files to render the UI. Finally link a website or domain to your SSP application. You can also create a SuiteApp (bundle) for your SSP application to share your customizations with other SuiteCommerce users.

To build an SSP application:

1. Turn on the required features in NetSuite. For more information, see [Enabling Required Features for Using SSP Applications](#).
2. Write the SuiteScripts that will implement your web store customization, including .ssp files to render the UI and .ss files to handle requests. For examples of these file types, see [Sample SSP Application Code](#).
3. Create and assemble other assets needed for your SSP application.
4. Create an SSP Application record in NetSuite. As part of this task, you create the file cabinet folder where the SSP application will be hosted. See [Creating an SSP Application Record](#).
5. Upload your SuiteScripts and other required assets to the file cabinet folder listed in the SSP Application record. See [Uploading SSP Application Files to the File Cabinet](#).
6. Edit the SSP Application record to define supported touch points where the SSP application will integrate with the web store. See, [Selecting Supported Touch Points](#).
7. Link the website to the SSP application in your NetSuite account. See [Linking Your Website to an SSP Application](#).
8. If you are working with multiple SSP applications in a single account, determine which one should be executed first. See [Changing SSP Application Precedence](#).

Debugging and Bundling

After you have created an SSP application, you can:

- Debug your web store customization by running your SuiteScripts and reviewing entries in the Script Execution Log on the SSP Application record. See [Debugging an SSP Application](#). (Note that use of the SuiteScript Debugger is not currently supported for SSP application scripts.)
- Use SuiteBundler to package your web store customization for installation in another NetSuite account. See [Bundling an SSP Application](#).

SSP Applications Compared to Customization with SuiteScript

The following tables compare SSP application capabilities with Suitelets, RESTlets, and Scriptable Cart:

- Presentation
- Services
- Authentication and Authorization

Deployment

KEY	
Symbol	Definition
S	Supported
U	Unsuported
—	Not Recommended
O	Optional

Presentation

	Suitelet	RESTlet	SSP Applications		Scriptable Cart
			.SSP	.SS	
Static HTML	S	X	S	X	—
UI Objects	S	X	—	—	X
Dynamic HTML	S	X	S	S	—
Standard Library Tags	X	X	S	X	—

Services

	Suitelet	RESTlet	SSP Application		Scriptable Cart
			.SSP	.SS	
JSON for nlobj	—	S	—	—	X
HTTP Methods	POST/GET	GET/POST/PUT/DELETE	GET/—	GET/POST	—
nl APIs	S	S	S	S	S (only on SO form)
Ecommerce APIs	X	X	S	S	X

Authentication and Authorization

	Suitelet	RESTlet	SSP Application		Scriptable Cart
			.SSP	.SS	
Available Without Login	S	X	S		S
Recognized User (Cookies)	X	X	S		S
Shopper	X	X	S		S
User (Entity Record)	S	S	S		S
Authentication Mechanism	JSessionId	JSessionId/nlAuth	JSessionId		JSessionId
Audience	S	S	X		X
Execute As Admin	S	X	X		X

Deployment

	Suitelet	RESTlet	SSP Application		Scriptable Cart
			.SSP	.SS	
Deployment Model	One per script	One per script	One per folder		many per form or Record Type
Named URLs	X	X	S		X
Touch Points to override NS pages	X	X	S		X
Touch Points to Extended Apps	S	S	X		X
Bundleable	S	S	S		S
Debugger	S	S	X		S

Enabling Required Features for Using SSP Applications

Before you can create, view, or link a website to an SSP Application record, the SuiteScript Server Pages feature must be enabled in your NetSuite account. Other features are also required. An account administrator or another user with the Enable Features permission can complete this setup.

To enable required features for SSP applications:

1. Go to Setup > Company > Enable Features.
2. On the **SuiteCloud** subtab, check the boxes for **Client SuiteScript**, **Server SuiteScript**, and **SuiteScript Server Pages**, and agree to the Terms of Service for each feature.
3. On the **Web Presence** subtab, check the boxes for Web Site and Host HTML Files.
4. Click **Save**.

Creating an SSP Application Record

The SSP Application record stores details about your website customization. This record enables you to group website assets, debug script files, and package your customization for use in other NetSuite accounts. You can define the following properties on the SSP application record:

- The folder in the NetSuite file cabinet where your website customization assets are stored.
- Supported touch points.
- URL root used in website links to website assets.

URL Root

The URL root, or base URL, for an SSP application specifies the path that points to the location of SSP application assets. By default, the URL root includes the application publisher and application name. However, the base URL may include as many path components as desired. The ability to set the base URL yourself provides flexibility and readability. NetSuite recommends that the base URL include at least one path component that distinguishes SSP application assets from other root assets in the domain.

For example, an SSP application named NextGenSite with an application publisher of SCA-NextGen, has the following default URL root: `/SCA-NextGen/NextGenSite/`. For testing purposes, you can access SSP application assets by navigating directly to the URL. For example, you can use the URL, `http://www.mydomain.com/SCA-NextGen/NextGenSite/index.ssp`.

Note: You must have the SuiteScript permission to access SSP application records.

The SSP Application Record

If you plan to create multiple SSP applications and publish them to other NetSuite accounts, you may need to create a new application publisher. If you already have an SSP application in your NetSuite account, you have the option of creating an SSP application record directly from the file cabinet.

For more information, read the following topics:

- [Creating an SSP Application Record from the File Cabinet](#)
- [Viewing and Adding Application Publishers](#)
- [Creating an SSP Application Folder](#)

To get started creating an SSP application, go to Customization > Scripting > SSP Applications > New. If you are creating the first SSP application in your NetSuite account, an SSP application folder is required. You can create the SSP application folder as you create the SSP application record.

The screenshot shows the 'SSP Application' form in NetSuite. It includes fields for NAME, ID, APPLICATION FOLDER, URL ROOT, DESCRIPTION, STATUS, LOG LEVEL, OWNER, and an INACTIVE checkbox. There are also tabs for Scripts and Supported Touch Points, and a Libraries section at the bottom. Numbered callouts (1-10) point to specific fields: 1 points to NAME, 2 to ID, 3 to APPLICATION FOLDER, 4 to URL ROOT, 5 to DESCRIPTION, 6 to STATUS, 7 to LOG LEVEL, 8 to OWNER, 9 to the INACTIVE checkbox, and 10 to the Scripts tab.

The following table describes each field on the SSP application form.

Step	Description
1	Enter a Name for the SSP application. This name is like a project name, and is used in NetSuite lists, for example, on the SSP Applications list page, and in the Bundle Builder. This name is not directly visible to web store users.
2	(Optional) Enter an ID . This identifier is used for scripting purposes, for the SSP Application record to be called by a SuiteScript. If you do not enter a value, a default is provided, <code>webapp###</code> . This value cannot be edited after the SSP Application record is saved.

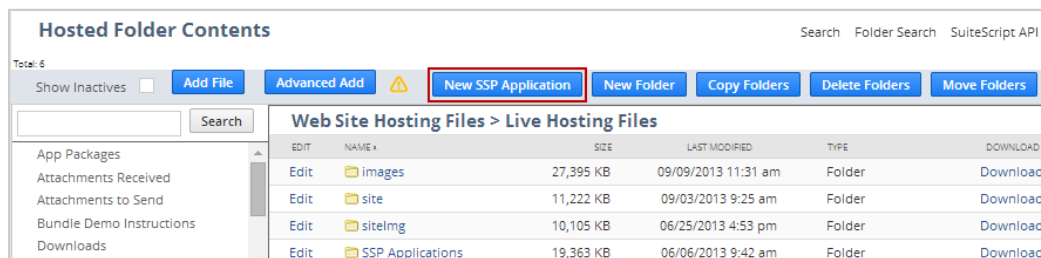
Step	Description
3	Select an Application Folder , or create a new folder. The list includes SSP Application folders in the WebSite Hosting Files Folder. For more information, see Creating an SSP Application Folder .
4	Review the URL Root and edit as desired. This is the base URL that is used to link to the SSP application from the web store and may be visible to web store users.
5	(Optional) Enter a Description . This is a description of the SSP application. The description is only displayed internally in NetSuite.
6	Select a Status . Leave as the default of Testing while you are creating and debugging your SSP application. Set to Released after you have completed the SSP application and linked it to your website.
7	(Optional) Select a Log Level . The value in this field determines the type of messages written to the script execution log for the SSP application. See Debugging an SSP Application .
8	(Optional) Select an Owner . Sets the owner of the SSP application. Default is the currently logged in user. After an SSP application is created, only the owner of the SSP application can modify it. The owner is different from the application publisher, as the owner refers to the individual user who is allowed to modify the SSP application record.
9	Add library script files on the Scripts subtab. These are external files that are called from the SSP application's script files. <ul style="list-style-type: none"> Files in the SuiteBundles, SuiteScripts, and Web Site Hosting Files folders of your file cabinet can be selected as library files. After you add a library file to an SSP application, it can be referenced by any of the .ss or .ssp files in the SSP application. For more information, see Uploading SSP Application Files to the File Cabinet .
10	Select Supported Touch Points . These are the pages where website visitors can interact with the SSP application. For more information, see Selecting Supported Touch Points .

Creating an SSP Application Record from the File Cabinet

If at least one SSP Application folder already exists in your file cabinet, you can create a new SSP Application record directly from the file cabinet.

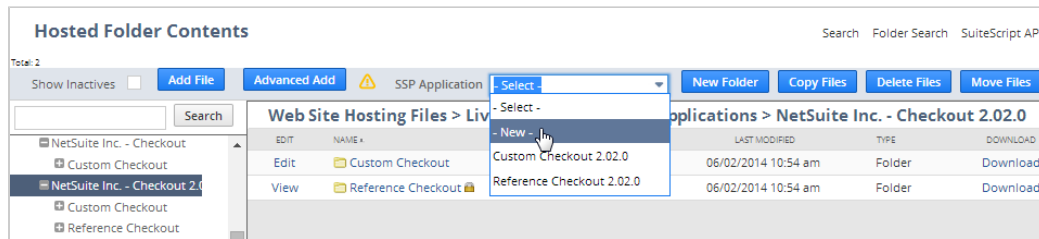
To navigate to the file cabinet for the purpose of creating an SSP application, go to Setup > SuiteCommerce Advanced > Web Site Hosting Files.

- You can select a website hosting folder and click **New SSP Application** to select an application publisher, and then continue on to create a new SSP Application record.



OR

- You can select an Application Publisher folder and use the dropdown list to create a new SSP Application record for that publisher.



Viewing and Adding Application Publishers

The application publisher identifies the creator of an SSP application. SSP applications are organized by application publisher. The default application publisher setting is your NetSuite account name. You can create multiple application publishers in your NetSuite account, but the name must be unique across all NetSuite accounts.

To view a list of application publishers, go to Setup > Company > Application Publishers (Admin). You can create a new application publisher from the same list page.

To create a new application publisher

1. Go to Setup > Company > Application Publishers (Admin).
2. Click **New Application Publisher**.
3. Enter a **Name** for the application publisher.



Important: An application publisher must be unique across all NetSuite accounts. If you enter a name that is already in use, an error is displayed.

4. Click **Save**.

The application publisher appears as a component in the URL root that points to the SSP application and all its assets. The SSP application folder path includes a subfolder named for the application publisher and a subfolder with the application name. (The default can be edited.)

The SSP application folder path has a format like the following:

<HTML Hosting Root>: /SSP Applications/<Application Publisher>/<Application Name>

After creating an application publisher, you can move on to [Creating an SSP Application Record](#).

Creating an SSP Application Folder

An SSP application is hosted from a web site hosting root SSP application folder. All of the SSP application scripts, libraries, images and other assets are contained in the SSP application folder and subfolders within the SSP application folder. A separate SSP application folder exists for each SSP application in your NetSuite account.

This organizational structure supports the coexistence of multiple customizations for a particular web store touch point, and a unique name space for each publisher. Each customization can be implemented separately. Different customizations do not collide with or overwrite one another.


You can create an SSP application folder directly from the SSP application record, or from the Web Site Hosting Files Folder in the file cabinet. An SSP application folder is required to store assets for each

SSP application in your account. The hierarchy of SSP Application folders is organized by application publishers.

You can create a new application publisher while creating the SSP application folder. The default value for application publisher is the same as your NetSuite company name. Alternatively, you can create an application publisher from the Application Publisher list page. For more information, see [Viewing and Adding Application Publishers](#).

To create an SSP application folder:


1. If this is your first SSP application, you must create a new application folder:
 - a. To create an SSP application folder from the SSP application record, click the + sign next to the **Application Folder** dropdown.
 - b. To create an SSP application folder from the file cabinet, go to Web Site Hosting Files > Live Hosting Files > SSP Applications. Click **New SSP Application**.

 **Note:** Create the SSP application folder in the hosting root that stores all the assets for your website. This can be the Live Hosting Files folder, the Staging Hosting Files folder, or a different folder that you create.

2. In the SSP Application Folder popup, select an **HTML Hosting Root**.
The list includes the folders in the Web Site Hosting Files folder in the file cabinet. (You can go to the file cabinet and create a new folder if necessary.)

3. Select an **Application Publisher**.

To create a new application publisher, click the + sign, enter the application publisher, and then click Save.

 **Note:** An application publisher must be unique across all NetSuite accounts.

4. Enter an **Application Name**. The application name becomes the name of the folder where all SSP application assets are stored. By default, this application name is included in the base URL that is visible to web store users. This application name can be different from the name on the SSP Application record, but it is desirable to keep them consistent to avoid confusion.
5. Click **Save**.

Notice the path for the SSP application folder. It uses the following format:

<HTML Hosting Root>: /SSP Applications/<Application Publisher>/<Application Name>

Uploading SSP Application Files to the File Cabinet

After [Creating an SSP Application Record](#), you can upload all of the scripts, images, and other assets for your web store customization to SSP application folder.

To upload assets to an SSP application record:

1. Go to Setup > SuiteCommerce Advanced > Web Site Hosting Files.
2. Select the SSP application folder you want to work with.
 - Create new folders as desired. For example, you might create subfolders in your SSP application folder for style sheets, images, javascript, and SuiteScript.
 - Add files as desired.
 - For more information about uploading files to the file cabinet, see the help topic [Working with the File Cabinet](#).

Note: For SuiteScript developers, notice that you need to upload your .ssp and .ss files to your SSP Applications folder. You should not upload these files to the SuiteScript folder in the file cabinet.

Using the include Tag instead of Library Files

For .ssp files, NetSuite recommends that you use the **include** tag to reference scripts rather than listing them as library script files. Use one of the following formats:

```
<%%include file="file"%>, or <%%include file="/path"%>
```

When the path begins with /, it is always relative to the root folder of the SSP application; otherwise it refers to a file in the same folder as the referring asset. You cannot use the **include** tag to reference files outside of the SSP application.

Note: All SSP applications are available without login. The ability to configure this option, which can be set in a check box on other NetSuite script records, is not currently available for the SSP Application record.

Applying Permissions to SSP Application Files

Administrators and users with privileges to use SuiteScript can set permissions on .ssp and .ss file records associated with an SSP application. With enhanced permissions enabled, a script can be triggered by an action performed by the users you specify. This enhanced permission capability is useful for creating website customizations. For example, you can create a personalized user interface for customers on your e-commerce website. A script can be triggered when a customer navigates to the script page on your shopping domain.

Note: Permissions can be applied only to SuiteScript files and SSP files within an SSP application folder.

To set permissions on an .ssp or .ss file:

1. Go to a folder in an SSP application. For example, Web Site Hosting Files > Live Hosting Files > SSP Applications > My SSP Application > My Script Files.
2. Click **Edit** next to the .ssp or .ss file you want to modify.
3. Click the **Permissions** subtab.
4. Check the **Enable** box.
5. In the **Execute as Role** list, select the role with which the script should be executed. When the script is executed, the outcome is only limited by the permissions associated with the role you select here. The script has access to the same permissions as the role you select.

Note: The Administrator role cannot be used.

6. Select individuals who can trigger the script to run:
 - a. (Optional) You can allow the SS or SSP script executable on the web store for all visitors. Check the **Run Script Without Login** box, or check the **Roles Select All** box to If you want to select specific roles or other criteria, then clear these boxes.

- b. (Optional) You can select a specific audience with permissions to execute the script. Select who can execute the script. You can choose more than one option in the following multi-select boxes:
 - Roles
 - Departments
 - Subsidiaries
 - Groups
 - Employees
 - Partners

7. Click **Save**.

When you configure permissions for an SSP or SS file, consider creating a custom role to use in the **Execute As Role** list. The custom role that you create should have exactly the privileges needed for your script to do what is intended, and nothing more. Typically, this script file permission capability is used for website customization.

Selecting Supported Touch Points

After [Uploading SSP Application Files to the File Cabinet](#), you can select supported touch points on the SSP Application record. Touch points point to files in the file cabinet. After selecting supported touch points on the SSP application record, you link your site or domain to one or more of the supported touch points by defining them on site and domain records.

For each supported touch point, NetSuite links to a page hosted by the SSP application. A single SSP application can be integrated at multiple touch points. You can define one or more supported touch points on the SSP Application record by associating each touch point with a website URL. For each URL, you have the option of defining parameters.

Supported touch points include the following:

- Log In
- Log Out
- Proceed to Checkout
- Register
- View Cart
- View Customer Center
- View Homepage

If you need to remove touch points from an SSP application or from a website, see [Removing Touch Points](#).

To select supported touch points on an SSP application:

1. Go to Setup > SuiteCommerce Advanced > SSP Applications.
2. Click **Edit** next to an SSP application.
3. Click the **Supported Touch Points** subtab.
4. Select each touch point in the **Name** column. The **Entry Page** column shows the URL that points to assets in the SSP application.

Supported Touch Points			
NAME *	ENTRY PAGE *	URL	PARAMETERS
Proceed to Checkout	/NetSuite Inc. - Checkout 2.04.0/Reference Checkout/index.ssp	/index.ssp	is=checkout
Register	/NetSuite Inc. - Checkout 2.04.0/Reference Checkout/index.ssp	/index.ssp	is=register
Log In	/NetSuite Inc. - Checkout 2.04.0/Reference Checkout/index.ssp	/index.ssp	is=login
View Cart	/NetSuite Inc. - Checkout 2.04.0/Reference Checkout/cart.ssp	/cart.ssp	

Note the following:

- When installing a reference implementation, supported touch points are displayed by default.
- If you use the website preference, Password Protect Entire Site, then the Log In touch point must be associated with an .ssp file. Other file types, such as HTML will not display in the website when used in this context.

- In the **Entry Page** field, enter or select the page to be inserted at the selected entry point.

Note: If you use the website preference, Password Protect Entire Site, then the Log In touch point must be associated with an .ssp file. Other file types, such as HTML will not display in the website when used in this context.

- Enter any URL parameters to be used.

All available touch points are displayed in the list. Note that in SuiteCommerce Advanced, some SSP applications are not compatible with every touch point. For more information about touch points available for specific reference implementations, see the help topic [Core SuiteCommerce Advanced Developer Tools](#).

After you have selected touch points of the SSP application, you can link the SSP application to your website. For more information, see [Linking Your Website to an SSP Application](#).

Removing Touch Points

You can remove touch points from the Web Site Setup page, from the Domains page, or from an SSP application record.

To remove touch points linked to a site:

- Go Setup > SuiteCommerce Advanced > Setup Web Site.
- Click the **Touch Points** subtab.
- Here, you can remove the touch point from the list.

To remove touch points linked to a domain:

- Go Setup > SuiteCommerce Advanced > Domains.
- Click the Edit link for the domain you want to modify on the list.
- Click the **Touch Points** field.
- Click the Touch Points Set button.

This opens the touch points in a new window.

- Here, you can remove the touch point from the domain.

Note that removing a touch point from a website or domain does not delete the touch point from the SSP application record. You can use the same touch point later.

To remove a touch point from an SSP application record:

1. Go to Customization > Scripting > SSP Applications.
2. Click **Edit** next to the SSP Application you want to modify.
3. Click the **Supported Touch Points** subtab.
4. Here, you can remove the touch point from the list.

Note that removing a touch point from the SSP application also removes that touch point from the Web Site Setup page. A touch point must exist on an SSP application for it to be accessible from the Web Site Setup page.

Linking Your Website to an SSP Application

Your website is linked to an SSP application when touch points are defined on the Web Site Setup page. After linking the website to an SSP application a URL is generated that points to assets in the SSP application for each supported touch point. You can use all the touch points supported by an SSP application, or only some of them.

You can also link a domain to an SSP application. In this way, your website customization is associated with a specific domain. For more information, see [Linking a Domain to an SSP Application](#).

To link your site to an SSP application:

1. Go to Customization > Scripting > SSP Applications.
2. On the list page, click **Edit** next to the SSP application you want to use on your website.
3. Verify that some **Supported Touch Points** are selected. For more information, see [Selecting Supported Touch Points](#).
4. Click the arrow in the **Save** button. Choose **Save & Link to Site**.
5. Select a site on the **Link Site To SSP Application** page.



Note: You can only link an SSP application to a website that has the same hosting root as the SSP application. For example, if you created an SSP application in the Staging Hosting Files folder, then you cannot link that SSP application to a website in the Live hosting root.

6. Click **Save**.

The touch points selected on the SSP Application record are listed on the Touch Points subtab of the Web Site Setup page, indicating the SSP application is live on the website. Note that you must complete additional setup tasks when installing a reference implementation. For more information, see the help topic [Core SuiteCommerce Advanced Developer Tools](#).



Important: NetSuite recommends that you first try linking your website to an SSP application in a sandbox account or another test site. If you do not have a test site, you can set up a local test site by modifying the hosts file on your computer. For more information, see the help topic [Editing the Hosts File](#).

Linking a Domain to an SSP Application

You can link a domain to supported touch points to create a unique web store experience for that domain. For example, you may want a certain look and feel for an entry point to your site using a certain domain.

To link a domain to an SSP application:

1. Follow Steps 1 – 4 in [Linking Your Website to an SSP Application](#).
2. Click **Save and Link to Domain**.
3. Select a domain on the **Link Domain To SSP Application** page.

Note: You can only select a domain that is pointing to the same hosting root as the SSP application. For example, you cannot deploy an SSP application created in the Staging Hosting files folder to a domain pointing to the Live Hosting Files folder.

4. Click **Save**.

Alternatively, you can click the Link to Domain button when the SSP application record is in view mode. To verify the domain is linked to the SSP application, visit the Domains subtab on the Web Site Setup page to confirm that you see the same supported touch points that exist on the SSP application.

See also, [Selecting Supported Touch Points](#), and [Removing Touch Points](#).

Changing SSP Application Precedence

SSP applications allow you to store multiple website customizations in one account, including customizations for the same web store touch point. When a URL refers to an asset in multiple SSP applications, a system of precedence determines which SSP application handles a particular request.

Precedence for SSP Application Assets

Each SSP application has a precedence sequence number, such as 1, 2, 3, 4, and so on. The lower the precedence number, the higher the application precedence. Files within an SSP application of higher precedence are used first.

By default, newer SSP applications take precedence over existing SSP applications when added to an account. However, you can change the precedence of any SSP application in NetSuite.

For example, you install an SSP application. Because this is the latest addition, it bears a precedence of 1 by default. Any files within this application take a higher precedence over files within SSP application folders with a precedence of 2, 3, 4, and so on. You can configure NetSuite to change this precedence at any time.

To change SSP App precedence:

1. Go to Customization > Scripting > SSP Applications.
2. Click the **Change Precedence** button.

SSP Applications

Change Precedence

New SSP Application

FILTERS

Show Inactives

EDIT VIEW	PRECEDENCE	NAME	FROM BUNDLE	ID	URL ROOT	APPLICATION PUBLISHER	HOSTING ROOT	APPLICATION FOLDER
Edit View	1	Custom My Account 1.03.0	53049	webapp_ns_cust_myaccount_1_03_0	/myaccount-1-03-0	NetSuite Inc. - My Account 1.03.0	Live Hosting Files	Web Site Hosting Files : Live Hosting Files : SSP Applications : NetSuite Inc. - My Account 1.03.0 : Custom My Account
Edit View	2	Reference My Account 1.03.0	53049	webapp_ns_myaccount_1_03_0	/myaccount-1-03-0	NetSuite Inc. - My Account 1.03.0	Live Hosting Files	Web Site Hosting Files : Live Hosting Files : SSP Applications : NetSuite Inc. - My Account 1.03.0 : Reference My Account

3. Change numbers in the **Precedence** column as desired and click **Submit**.

Sample SSP Application Code

The following examples are intended to provide you with a basic idea of how .ssp and .ss files are structured.

- [Sample_Cart.ssp](#)
- [Cart_WithService.ssp](#)
- [UpdateQuantity.ss](#)
- [AddCartItem.ss](#)
- [AddCustomerAddress.ss](#)
- [Example .ss File Code that Accesses a NetSuite Record](#)
- [ItemOption.ss](#)



Note: For code samples that integrate with PayPal Express and 3DSecure, see [Integration with Third Party Checkout Providers](#).

Sample_Cart.ssp

The following code creates a very basic View Cart touchpoint:

```
<html>
<head><%=getPageFullHead()%>
<!--add css, js packages here -->
<link rel="stylesheet" type="text/css" href="/checkout/css/style2.css" >
<% var order = nlapiGetWebContainer().getShoppingSession().getOrder().getFieldValues({'items':
['name', 'salesdesc', 'quantity', 'rate', 'amount'], 'summary': ''});%>
<script type="text/javascript"> var nlShoppingOrder = <%=JSON.stringify(order)%>; </script>
<script type="text/javascript">
    function createCartItemRow(nlCartItem, itemAttributes)
    {
        var row = document.createElement('tr');

        for (var i = 0; i < itemAttributes.length; i++)
        {
            row.appendChild(createCartItemCell(nlCartItem[itemAttributes[i]]));
        }

        return row;
    }
    function createCartItemCell(nlCartItemAttr)
    {
        var cell = document.createElement('td');
        cell.setAttribute('class', 'texttable');
        var txtCell = document.createTextNode(nlCartItemAttr);
        cell.appendChild(txtCell);
        return cell;
    }

    function createCartTableHeader(headerItems)
    {
        var theader = document.createElement('thead');
        theader.setAttribute('id', 'carttableheader');
```

```

        for (var i = 0; i < headerItems.length; i++)
        {
            theader.appendChild(createCartHeaderCell(headerItems[i]));
        }
        return theader;
    }
    function createCartHeaderCell(nlCartHeaderItem)
    {
        var cell = document.createElement('th');
        var txtCell = document.createTextNode(nlCartHeaderItem);
        cell.appendChild(txtCell);
        return cell;
    }

    function createShoppingCart()
    {
        var headerItems = ['Name', 'Description', 'Quantity', 'Rate', 'Amount'];
        var itemAttributes = ['name', 'salesdesc', 'quantity', 'rate', 'amount'];
        var shoppingCart = document.createElement('table');
        shoppingCart.setAttribute('id', 'carttable');
        shoppingCart.setAttribute('width', '100%');

        var theader = createCartTableHeader(headerItems);
        shoppingCart.appendChild(theader);

        var tbody = document.createElement('tbody');

        if (nlShoppingOrder && nlShoppingOrder.items)
        {
            for (var i = 0; i < nlShoppingOrder.items.length; i++)
            {
                tbody.appendChild(createCartItemRow(nlShoppingOrder.items[i], itemAttributes));
            }
        }

        shoppingCart.appendChild(tbody);
        document.getElementById('mainContents').appendChild(shoppingCart);
    }
}

</script>
</head>
<body onLoad="createShoppingCart()">

<table cellpadding=0 cellspacing=0 border=0 width=100%><NLPAGETOP></table>

<table>
<%nlapiGetWebContainer().getPageGenerator().setSelectedTab(3);%>

<div id="mainContents"></div>

</table>
</body>
</html>

```


Cart_WithService.ssp

The following code creates a View Cart touchpoint that references an UpdateQuantity backend function. To work, this code requires the UpdateQuantity.ss file.

```
<html>
<head>
    <%=getPageFullHead()%>
<script language="JavaScript">

var xmlhttp = new XMLHttpRequest();

/*
 * Dynamically build out the cart header
 */
function buildCartHeader()
{
    var tHead = cartTable.createTHead();
    var row, cell;

    row = tHead.insertRow(-1);
    tHead.setAttribute("bgColor","lightskyblue");

    cell = row.insertCell(-1);
    cell.align = "center";
    cell.style.fontWeight = "bold";
    cell.innerHTML = 'Item';

    cell = row.insertCell(-1);
    cell.align = "center";
    cell.style.fontWeight = "bold";
    cell.innerHTML = 'QTY';

    cell = row.insertCell(-1);
    cell.align = "center";
    cell.style.fontWeight = "bold";
    cell.innerHTML = 'Rate';

    cell = row.insertCell(-1);
    cell.align = "center";
    cell.style.fontWeight = "bold";
    cell.innerHTML = 'Amount';
}

/*
 * Dynamically build out the cart
 */
function buildCartItems(items)
{
    var row, cell;

    // Insert table rows and cells into body
    for (var i=0; i<items.length; i++)
    {
```

```

    row = cartBody.insertRow(-1);
    cell = row.insertCell(-1);

    cell.innerHTML = items[i].name;

    cell = row.insertCell(-1);
    cell.innerHTML = items[i].quantity;

    cell = row.insertCell(-1);
    cell.innerHTML = items[i].rate;

    cell = row.insertCell(-1);
    cell.innerHTML = items[i].amount;
}

// Set the background color of the table body.
cartBody.setAttribute("bgColor", "white");
}

/*
 * Process result of AJAX call to update quantity
 */
function handleServiceResponse()
{
    if (xmlhttp.readyState == 4 && xmlhttp.status == 200)
    {
        // remove the cart contents
        document.getElementById('cartBody').innerHTML = '';

        // parse the JSON result into JavaScript objects,
        // and rebuild the cart
        var items = JSON.parse(xmlhttp.responseText);
        buildCartItems(items);
    }
}

/*
 * AJAX call to NetSuite service
 * Service updates the quantity on line 1
 */
function updateQuantity()
{
    xmlhttp.onreadystatechange=handleServiceResponse;

    xmlhttp.open('POST', 'updateQuantity.ss', true);

    xmlhttp.setRequestHeader('Content-type','application/x-www-form-urlencoded');

    try
    {
        xmlhttp.send('quantity=' +
document.getElementById('quantity').value);

```

```

    }
    catch (ex)
    {
        alert(ex);
    }
}

</script>
</head>
<body>

<table cellpadding=0 cellspacing=0 border=0 width=100%>
    <%=getPageTop()%>
</table>

<button id="updateqty" type="button" onclick="updateQuantity();">Update
QTY(line1)</button>
<input id="quantity" type="text"/>

<br><br>

<table id="cartTable" border="1" bgcolor="lightslategray">
    <tbody id="cartBody"></tbody>
</table>

<script language="JavaScript">
<%var order = nlapiGetWebContainer().getShoppingSession().getOrder();%>

// JSON.parse not used here since client script is generated similar to...
//     var items = '{"itemid":"Urban Dining Table","onlinecustomerprice":"1686.26"}'
// Setting a variable explicitly equal to a JSON string automatically converts to
//
objects
var items = <%=JSON.stringify(order.getItems())%>

buildCartHeader();
buildCartItems(items);

</script>

</body>
</html>

```

UpdateQuantity.ss

The following code contains the backend function for updating quantity in the Cart_WithService .ssp file:

```

function service(request,response)
{
    // get incoming http parameter

```

```

var quantity = request.getParameter('quantity');
nlapiLogExecution('DEBUG', 'quantity', quantity);

// get active order in web container
var order = nlapiGetWebContainer().getShoppingSession().getOrder();

// get the items out of the order
var items = order.items;

var orderItemId = items[0].orderid;
nlapiLogExecution('DEBUG', 'order item id - line 1', orderItemId);

var currentQuantityLine1 = items[0].quantity;
nlapiLogExecution('DEBUG', 'order curr qty - line 1', currentQuantityLine1);

var updatedItem = new Object();
updatedItem.orderitemid = orderItemId;
updatedItem.quantity = quantity;

order.updateItemQuantity(updatedItem);

var itemsJSON = JSON.stringify(order.getItems());

// send items back out as JSON,
response.setContentType('JSON');
response.writeLine(itemsJSON);
}

```

AddCartItem.ss

```

function service(request,response)
{
    var itemid = request.getParameter('itemid');
    var qty = request.getParameter('quantity');

    // create a plain object for the item
    var item = {internalid : itemid, quantity: qty}

    // Get a reference to the shopping order
    var order = nlapiGetWebContainer().getShoppingSession().getOrder();

    // Add the item to cart
    try
    {
        var returnval = order.addItem(item);

        response.writeLine(JSON.stringify(returnval));
    }
    catch (e)
    {
        var e2 = nlapiCreateError(e);
        // Do error handling here
    }
}

```

```
}

```

AddCustomerAddress.ss

```
function service(request,response) {
    /*var addressee = request.getParameter('addressee');
    var addr1 = request.getParameter('addr1');
    var city = request.getParameter('city');
    var state = request.getParameter('state');
    var zip = request.getParameter('zip');
    var country = request.getParameter('country');
    var phone = request.getParameter('phone');

    var address = {
        addressee : addressee,
        addr1 : addr1,
        city : city,
        state : state,
        zip : zip,
        country : country,
        phone : phone,
        defaultshipping : 'T',
        defaultbilling : 'T'
    };*/

    // Or the object string can be passed from the request header
    var address = JSON.parse(request.getHeader('customerAddress'));

    // Get a reference to the customer
    var customer = nlapiGetWebContainer().getShoppingSession().getCustomer();

    try
    {
        // Add the address
        var status = customer.addAddress(address);

        var result = customer.getFieldValues();

        // return the customer data through response
        response.writeLine(JSON.stringify(result.customer));
    }
    catch (e)
    {
        var e2 = nlapiCreateError(e);
        // Do error handling here
    }
}
```

Example .ss File Code that Accesses a NetSuite Record

Code like the following can be used in an .ss file to access a NetSuite record rather than a shopping session object.

```
function service(request,response)
{
    var salesOrder = nlapiCreateRecord('salesorder', true);

    salesOrder.setFieldValue('customform', 88); // Cash Sale form.
    salesOrder.setFieldValue('orderstatus', 'B');
    salesOrder.selectNewLineItem('item');
    salesOrder.setCurrentLineItemValue('item', 'item', 6);
    salesOrder.setCurrentLineItemValue('item', 'quantity', 1);
    salesOrder.setCurrentLineItemValue('item', 'rate', 500);
    salesOrder.commitLineItem('item');
    nlapiSubmitRecord(salesOrder);
}
```



Important: Running an .ss file with code accessing a NetSuite record requires a preexisting NetSuite session. Running this code without first logging in to NetSuite results in permission errors. Running this type of code from the shopping domain also results permission errors, because shoppers are only recognized in the shopping domain.

ItemOption.ss

```
/*
http://www.mystore.com/test/itemOption.ss
*/

function service(request,response)
{
    try
    {
        var thesize = request.getParameter('size');
        var msg = request.getParameter('msg');

        if (thesize == null || msg == null)
        {
            throw "Parameters 'size' and 'msg' need to be included";
        }

        var order = nlapiGetWebContainer().getShoppingSession().getOrder();

        var itemid = '74';
        var qty = '8';
        var optionMap = { 'custcol_size':'2', 'custcol_message':'FIRST MESSAGE' };
        var itemMap = { 'internalid' : itemid, 'quantity': qty, 'options' : optionMap };
        order.addItem(itemMap);
    }
}
```

```

        response.writeLine("Finished adding new item");
        writeOrder(order);

        var itemArray = order.getFieldValues(['items']).items;
        response.writeLine("Number of order lines "+itemArray.length);

        // Get the order item ID from the order
        var theitemid = itemArray[0].orderid;
        response.writeLine("The order item ID is "+theitemid);

        // Now update the item options
        var thesize = request.getParameter('size');
        var msg = request.getParameter('msg');

        optionMap = { 'custcol_size' : thesize, 'custcol_message':msg};
        order.setItemOptions(theitemid, optionMap);
        response.writeLine("Updated item option");
        writeOrder(order);

        /*
        response.writeLine("Clearing the cart");
        order.removeAllItems();
        */

    }
    catch (ex) {
        response.writeLine(ex);
    }
}

function writeOrder(order)
{
    var orderDetails = order.getFieldValues(['items']);
    var orderItems = new Array();
    for (var i = 0; i < orderDetails.items.length; i++)
    {
        var orderItem = {};
        orderItem.internalid = orderDetails.items[i].internalid;
        orderItem.orderitemid = orderDetails.items[i].orderid;
        orderItem.quantity = orderDetails.items[i].quantity;
        orderItem.rate = orderDetails.items[i].rate;
        orderItem.amount = orderDetails.items[i].amount;
        orderItem.name = orderDetails.items[i].name;
        orderItem.options = orderDetails.items[i].options;
        orderItems[i] = orderItem;
    }
    response.writeLine(JSON.stringify(orderItems));
}

```

Known Issue with Internet Explorer and Plain Text Content

If the content type for a file is set to text/plain in the HTTP header and the file name has an .ssp or .ss extension (defined in either the response header or the URL), Internet Explorer may attempt to save the file rather than display it inline.

This is a known issue with Internet Explorer 9 and may also occur with previous versions of Internet Explorer. This issue does not occur with files that have HTML rather than text content.

To ensure that Internet Explorer displays the contents of a file inline, you should use `response.setContentType()` to rename the file with a .txt extension, as in the following example:

```
function service(request, response)
{
    response.setContentType('PLAINTEXT', 'my_SSP.txt', 'inline');
    response.writeLine('hey');
}
```


Integration with Third Party Checkout Providers

For integration with third party checkout providers such as PayPal Express and 3D secure, you need to create request handlers (.ss files) for these providers, and call these handlers from your frontend page (.ssp file). The request handlers should use `proceedToCheckout()`, as documented in [ShoppingSession Methods](#). This method must pass `checkoutsettings` JSON object values as parameters. Note that this method can only be called from a secure (https) scheme. For more details, see:

- [Integrating with PayPal Express](#)
- [Integrating with 3D Secure](#)

Integrating with PayPal Express

To integrate SSP application checkout with PayPal Express, you must do the following for the View Cart and Proceed to Checkout touch points:

- Get the `checkoutsettings` JSON object by calling `nlapiGetWebContainer().getShoppingSession().getSiteSettings()`.
- Ensure that the `checkout.paypalexpress.available` field is set to T in the `checkoutsettings` object.
- Get the image URL for PayPal Express from the `checkout.paypalexpress.imageurl` field in the `checkoutsettings` object
- Add the Paypal Express Checkout button to your touch point .ssp file and provide a handler, in an .ss file, that calls the `proceedToCheckout()` method on the shopping session, to integrate with the PayPal Express checkout flow.

Values for `checkoutsettings` fields should be passed as parameters for this method. You can set values for the following fields to customize the default PayPal Express checkout flow:

- **type** – set to `paypalexpress`
- **cancelurl** – the URL of redirect page when Cancel link is clicked at PayPal site
- **continueurl** – URL for redirect page when Continue link is clicked at PayPal site
- **createorder** – if value is set to T, Backend submits the current shopping order when Continue link is clicked; default is F

In addition to an .ss file, integration with PayPal Express requires client-side code to display PayPal Express information in the user interface. For examples of this code, see the following:

- [Sample .ss File for PayPal Express Integration](#)
- [Sample for Displaying PayPal Express Information](#)

Sample .ss File for PayPal Express Integration

The following code is in a sample `PaypalCheckout.ss` file:

```
function service(request,response)
{
    var returnval = null;

    try
    {
```

```

var shoppingSession = nlapiGetWebContainer().getShoppingSession();
var siteSetting = shoppingSession.getSiteSettings(['touchpoints']);

var viewcart = siteSetting.touchpoints.viewcart;
var homeurl = siteSetting.touchpoints.home;

var checkoutSetting = {type: 'paypalexpress', continueurl: homeurl, cancelurl : viewcart
t};

nlapiGetWebContainer().getShoppingSession().proceedToCheckout(checkoutSetting);
}
catch (e)
{
var nle = nlapiCreateError(e);
returnval = {status : 'error', reasoncode : nle.getCode(), message : nle.getDetails()};

response.writeLine(JSON.stringify(returnval));
}
}

```

Sample for Displaying PayPal Express Information

The following code can be used for PayPal Express integration. Note the use of the `getAbsoluteURL(domain, path)` method for the Shopping Session object. The absolute path is required to avoid errors for cross-domain links (links that go between shopping and checkout domains).

```

<table border=0>
<% var checkoutSettings = nlapiGetWebContainer().getShoppingSession().getSiteSettings(['checkou
t']);
var paypalUrl = nlapiGetWebContainer().getShoppingSession().getAbsoluteUrl("checkout", "sampleP
aypalCheckout.ss");
var proceedToCheckoutUrl = nlapiGetWebContainer().getShoppingSession().getAbsoluteUrl("checkout
", "sampleCheckout.ss");
<tr>
<td>
<form name='checkoutform' action='<%=proceedToCheckoutUrl%>' method='post'>
<input type="submit" name="placeorder" id="placeorder" value="Proceed To Checkout" class=
"nlbutton"/>
</form>
</td>
<td> </td>
<td >
<form name='paypalform' action='<%=paypalUrl%>' method='post'>
<input type='image' src='<%=checkoutSettings.checkout.paypalexpress.imageurl%>' class="nl
button"/>
</form>
</td>
</tr>
</table>

```

```
function service(request,response)
```

```

{
    var returnval = null;

    try
    {

        var shoppingSession = nlapiGetWebContainer().getShoppingSession();
        var siteSetting = shoppingSession.getSiteSettings(['touchpoints']);

        var viewcart = siteSetting.touchpoints.viewcart;
        var homeurl = siteSetting.touchpoints.home;

        var checkoutSetting = {type: 'paypalexpress', continueurl: homeurl, cancelurl : viewcart}
    ;

        nlapiGetWebContainer().getShoppingSession().proceedToCheckout(checkoutSetting);
    }
    catch (e)
    {
        var nle = nlapiCreateError(e);
        returnval = {status : 'error', reasoncode : nle.getCode(), message : nle.getDetails()};

        response.writeLine(JSON.stringify(returnval));
    }
}

```

Integrating with 3D Secure

To integrate your SSP application's checkout with 3D Secure, you must do the following for the Proceed to Checkout touch point:

In an .ss file, call the submit method on the order shopping object, and pass the following paymentauthorization object fields as [checkoutsettings](#) parameters:

- type - set to "threedsecure"
- noredirect - indicates whether platform returns a status object with required fields for 3D Secure instead of doing a redirect when 3D Secure authorization is required, default is F

Note the following:

 - If noredirect is set to F [threedsecure](#) , redirect goes to the default 3D Secure authorization page.
 - If noredirect is set to T when 3D Secure authorization is required, the submit method returns the order status as error, with a reason code of ERR_WS_REQ_PAYMENT_AUTHORIZATION, along with values for the paymentauthorization JSON object. You can use these values to handle the integration with 3D Secure.
- termurl - URL for client-side handler to process 3D Secure callback, to submit order with 3D Secure response parameters

In addition to the .ss file, 3D secure integration requires client-side code to display 3D Secure information in the user interface and to provide order handling. For examples of this code, see the following:

- [Sample .ss File for 3D Secure Integration](#)
- [Sample for Adding 3D Secure Frame in User Interface](#)

- Sample Client-Side Handler for 3D Secure Order Submission
- sampleShoppingLib.js
- sampleClientLib.js
- samplePaypalCheckout.ss

Sample .ss File for 3D Secure Integration

```
function service(request,response)
{
    var returnval = null;

    try
    {
        var shoppingSession = nlapiGetWebContainer().getShoppingSession();

        var shipMethod = shoppingSession.getOrder().getShippingMethod();

        if (!shipMethod || !shipMethod.shipMethod)
        {
            var shipMethods = shoppingSession.getOrder().getAvailableShippingMethods();
            nlapiLogExecution("DEBUG", "setShippingMethod", JSON.stringify(shipMethods));
            shoppingSession.getOrder().setShippingMethod(shipMethods[0]);
        }

        var orderHandlerUrl = shoppingSession.getAbsoluteUrl('checkout', 'samplePlaceOrder.ss');
        var orderSetting = {paymentauthorization:{type: 'threedsecure', noredirect : 'T', termurl
: orderHandlerUrl}};
        returnval = shoppingSession.getOrder().submit(orderSetting);
    }
    catch (e)
    {
        var nle = nlapiCreateError(e);
        returnval = {status : 'error', reasoncode : nle.getCode(), message : nle.getDetails()};
    }
    response.writeLine(JSON.stringify(returnval));
}
```

Sample for Adding 3D Secure Frame in User Interface

```
<html>
<head><%=getPageFullHead()%>
<!--add css, js packages here -->
<link rel="stylesheet" type="text/css" href="css/style2.css" >
<script language='JavaScript' type='text/javascript'>
try {
    if (window.parent != window && window.parent.document.getElementById('threedsecureframe') !=
null) { window.parent.location.href = window.location.href; }
    } catch(e){ }
</script>
<script type="text/javascript" src="lib/sampleShoppingLib.js"></script>
<script type="text/javascript" src="lib/sampleClientLib.js"></script>
</head>
<body onLoad="loadShoppingCart();">
```

```

<table cellpadding=0 cellspacing=0 border=0 width=100%><NLPAGETOP></table>

<%nlapiGetWebContainer().getPageGenerator().setSelectedTab(3);%>
<table width="100%">
<tr>
<td><div id="mainContents"></div></td>
</tr>
</table>
<table border=0>
<% var checkoutSettings = nlapiGetWebContainer().getShoppingSession().getSiteSettings(['checkou
t']);
var paypalUrl = nlapiGetWebContainer().getShoppingSession().getAbsoluteUrl("checkout", "sampleP
aypalCheckout.ss");
<tr>
<td>
<input type="submit" name="placeorder" id="placeorder" value="Submit" style="" class="nlbutton"
onclick="placeOrder()">
</td>
<td>&nbsp;</td>
<td >
<form name='paypalform' action='<%=paypalUrl%>' method='post'>
<input type='image' src='<%=checkoutSettings.checkout.paypalexpress.imageurl%>' class="nlbutton
"/>
</form>
</a>
</td>
</tr>
</table>
<div id="paymentauthenticator"></div>
</body>
</html>

```

Sample Client-Side Handler for 3D Secure Order Submission

```

function placeOrderCallBack(req)
{
    var responseObj = JSON.parse(req.responseText);
    if (responseObj.statuscode && responseObj.statuscode == "error")
    {
        // Order is not successful since payment authorization is required
        if (responseObj.reasoncode && responseObj.reasoncode == "ERR_WS_REQ_PAYMENT_AUTHORIZATI
ON")
        {
            if (responseObj.paymentauthorization)
            {
                // grab the frame html for threedsecure and inject it into the page. this will
                trigger the authorization flow
                document.getElementById('paymentauthenticator').innerHTML = responseObj.payment
                authorization.servicehtml;
                var placeOrderBtn = document.getElementById('placeorder');
                if (placeOrderBtn)
                {
                    placeOrderBtn.parentNode.removeChild(placeOrderBtn);
                    var paypalBtn = document.getElementById('paypalcheckout');

```

```

        if (paypalBtn)
            paypalBtn.parentNode.removeChild(paypalBtn);
    }
}
}
}

```

```

function service(request,response)
{
    var returnval = null;

    try
    {
        var shoppingSession = nlapiGetWebContainer().getShoppingSession();
        var orderHandlerUrl = shoppingSession.getAbsoluteUrl('checkout', 'samplePlaceOrder.ss');
        var orderSetting = {paymentauthorization:{type: 'threedsecure', noredirect : 'T', termu
rl : orderHandlerUrl}};
        returnval = shoppingSession.getOrder().submit(orderSetting);    }
    catch (e)
    {
        var nle = nlapiCreateError(e);
        returnval = {status : 'error', reasoncode : nle.getCode(), message : nle.getDetails()};
    }
    response.writeLine(JSON.stringify(returnval));
}

```

```

function placeOrderCallBack(req)
{
    var responseObj = JSON.parse(req.responseText);
    if (responseObj.statuscode && responseObj.statuscode == "error")
    {
        // Order is not success since payment authorization is required
        if (responseObj.reasoncode && responseObj.reasoncode == "ERR_WS_REQ_PAYMENT_AUTHORIZATION")
        {
            if (responseObj.paymentauthorization)
            {
                // grab the frame html for threedsecure and inject it into the page. this will trigger the authorization flow
                document.getElementById('paymentauthenticator').innerHTML = responseObj.paymentauthorization.servicehtml;
                var placeOrderBtn = document.getElementById('placeorder');
                if (placeOrderBtn)
                    placeOrderBtn.parentNode.removeChild(placeOrderBtn);
                var paypalBtn = document.getElementById('paypalcheckout');
                if (paypalBtn)
                    paypalBtn.parentNode.removeChild(paypalBtn);
            }
        }
    }
}

```

sampleShoppingLib.js

```

function createCartItemRow(nlCartItem, itemAttributes)
{
    var row = document.createElement('tr');

    for (var i = 0; i < itemAttributes.length; i++)
    {
        row.appendChild(createCartItemCell(getCartItemAttributeText(itemAttributes[i], nlCartItem
[itemAttributes[i]])));
    }

    return row;
}

function getCartItemAttributeText(attrName, attrValue)
{
    if (attrName === 'options')
    {
        var attrText = '';
        if (attrValue)
        {
            for (var i = 0; i < attrValue.length; i++)
            {
                var itemOption = attrValue[i];
                attrText += itemOption.name + ':' + itemOption.displayvalue + '\n';
            }
        }
        return attrText;
    }
    else
        return attrValue;
}

function createCartItemCell(nlCartItemAttr)
{
    var cell = document.createElement('td');
    cell.setAttribute('class', 'texttable');
    var txtCell = document.createTextNode(nlCartItemAttr);
    cell.appendChild(txtCell);
    return cell;
}

function createCartTableHeader(headerItems)
{
    var theader = document.createElement('thead');
    theader.setAttribute('id', 'carttableheader');
    for (var i = 0; i < headerItems.length; i++)
    {
        theader.appendChild(createCartHeaderCell(headerItems[i]));
    }
    return theader;
}

function createCartHeaderCell(nlCartHeaderItem)
{
    {

```

```

    var cell = document.createElement('th');
    var txtCell = document.createTextNode(nlCartHeaderItem);
    cell.appendChild(txtCell);
    return cell;
}

function createShoppingCart(nlShoppingOrder)
{
    var headerItems = ['Name', 'Description', 'Options', 'Quantity', 'Rate', 'Amount'];
    var itemAttributes = ['name', 'salesdesc', 'options', 'quantity', 'rate', 'amount'];
    var shoppingCart = document.createElement('table');
    shoppingCart.setAttribute('id', 'carttable');
    shoppingCart.setAttribute('width', '100%');

    var theader = createCartTableHeader(headerItems);
    shoppingCart.appendChild(theader);

    var tbody = document.createElement('tbody');

    if (nlShoppingOrder && nlShoppingOrder.items)
    {
        for (var i = 0; i < nlShoppingOrder.items.length; i++)
        {
            tbody.appendChild(createCartItemRow(nlShoppingOrder.items[i], itemAttributes));
        }
    }

    shoppingCart.appendChild(tbody);
    document.getElementById('mainContents').appendChild(shoppingCart);
}

function loadShoppingCart() {
    sendRequest('sampleGetOrderFields.ss', loadShoppingCartCallback);
    sendRequest('sampleGetCheckoutUrl.ss', loadCheckoutSettingCallback);
}

function loadCheckoutSetting() {
    sendRequest('sampleGetCheckoutUrl.ss', loadCheckoutSettingCallback);
}

function loadCheckoutSettingCallback(req) {
    var checkoutSetting = JSON.parse(req.responseText);
    var paypalBtn = document.getElementById('paypalcheckout');
    if (paypalBtn)
        paypalBtn.href = checkoutSetting.paypalexpress;
}

function loadShoppingCartCallback(req) {
    //render the items table
    var order = JSON.parse(req.responseText);
    createShoppingCart(order);
}

function placeOrder()
{

```



```

    sendRequest('samplePlaceOrder.ss', placeOrderCallBack);
}
function placeOrderCallBack(req)
{
    var responseObj = JSON.parse(req.responseText);
    if (responseObj.statuscode && responseObj.statuscode == "error")
    {
        // Order is not success since payment authorization is required
        if (responseObj.reasoncode && responseObj.reasoncode == "ERR_WS_REQ_PAYMENT_AUTHORIZATION")
        {
            if (responseObj.paymentauthorization)
            {
                // grab the frame html for threedsecure and inject it into the page. this will trigger the authorization flow
                document.getElementById('paymentauthenticator').innerHTML = responseObj.paymentauthorization.servicehtml;
                var placeOrderBtn = document.getElementById('placeorder');
                if (placeOrderBtn)
                    placeOrderBtn.parentNode.removeChild(placeOrderBtn);
                var paypalBtn = document.getElementById('paypalcheckout');
                if (paypalBtn)
                    paypalBtn.parentNode.removeChild(paypalBtn);
            }
        }
    }
}
}

```

sampleClientLib.js

```

function sendRequest(url,callback,postObjName, postData) {
    var req = createXMLHTTPObject();
    if (!req) return;
    var method = (postData) ? "POST" : "GET";
    req.open(method,url,true);
    req.setRequestHeader('User-Agent','XMLHTTP/1.0');
    if (postData) {
        req.setRequestHeader('Content-type','application/x-www-form-urlencoded');
        req.setRequestHeader(postObjName, JSON.stringify(postData));
    }
    req.onreadystatechange = function () {
        if (req.readyState != 4) return;
        if (req.status != 200 && req.status != 304) {
            // alert('HTTP error ' + req.status);
            return;
        }
        callback(req);
    }
    if (req.readyState == 4) return;
    req.send();
    //req.send(postData);
}

```

```

var XMLHttpRequestFactories = [
    function () {return new XMLHttpRequest();},
    function () {return new ActiveXObject("Msxml2.XMLHTTP");},
    function () {return new ActiveXObject("Msxml3.XMLHTTP");},
    function () {return new ActiveXObject("Microsoft.XMLHTTP");}
];

function createXMLHTTPObject() {
    var xmlhttp = false;
    for (var i=0;i<XMLHttpRequestFactories.length;i++) {
        try {
            xmlhttp = XMLHttpRequestFactories[i]();
        }
        catch (e) {
            continue;
        }
        break;
    }
    return xmlhttp;
}

```

samplePaypalCheckout.ss

```

function service(request,response)
{
    var returnval = null;

    try
    {
        var shoppingSession = nlapiGetWebContainer().getShoppingSession();
        var siteSetting = shoppingSession.getSiteSettings(['touchpoints']);

        var viewcart = siteSetting.touchpoints.viewcart;
        var homeurl = siteSetting.touchpoints.home;
        var checkouturl = siteSetting.touchpoints.checkout;

        var testUrl = shoppingSession.getAbsoluteUrl('/sampleGcoCheckout.ss');
        var testUrl2 = shoppingSession.getAbsoluteUrl('lib/sampleShoppoingLib.js');

        var paypalSetting = {type: 'paypalexpress', continueurl : checkouturl, cancelurl : viewcart};

        nlapiGetWebContainer().getShoppingSession().proceedToCheckout(paypalSetting);
    }
    catch (e)
    {
        var nle = nlapiCreateError(e);
        returnval = {status : 'error', reasoncode : nle.getCode(), message : nle.getDetails()};

        response.writeLine(JSON.stringify(returnval));
    }
}

```

Debugging an SSP Application

The creation of an SSP Application record enables you to write debugging details to the script execution log when SSP application scripts are run. These details are displayed on the Execution Log subtab of the SSP Application record.

Use the SuiteScript API `nlapiLogExecution(type, title, details)` to debug SSP applications. This API requires the following parameters:

- *type* {string} [required] - One of the following log types:
 - DEBUG (the default when Status is set to Testing)
 - AUDIT
 - ERROR
 - EMERGENCY
- *title* {string} [required] - A title used to organize log entries (max length: 99 characters). **Important:** The *title* argument is mandatory and a value must be specified. You cannot set *title* to *null* or to an empty string.
- *details* {string} [optional] - The details of the log entry (max length: 3999 characters)

For more details about this API, see the SuiteScript Help.



Note: Use of the SuiteScript Debugger currently is not supported for SSP application scripts.

Bundling an SSP Application

After you have created an SSP Application record and uploaded its assets to the file cabinet, you can use the Bundle Builder in NetSuite to package the SSP application into a SuiteApp for distribution to other NetSuite accounts. For more information on creating a bundle, see the help topic [Creating a Bundle with the Bundle Builder](#). Note that if you do not want users in target accounts to edit SSP application files, you can lock the SSP application as part of creating the bundle.

Additional notes on bundling SSP applications:

- SSP applications are listed in Step 3 of the Bundle Builder, where you select objects to include in the bundle.
- When you select an SSP application here, the SSP Application record and all of the files in the file cabinet application folder for the selected SSP application are included in the bundle.
- When the bundle is installed in a target account, the complete SSP Application record is copied to that account. Also, all files are copied to an application folder in the target account file cabinet, in the following location:

<Default Target HTML Hosting Root>: /SSP Applications/<Application Publisher>/<Application Name>