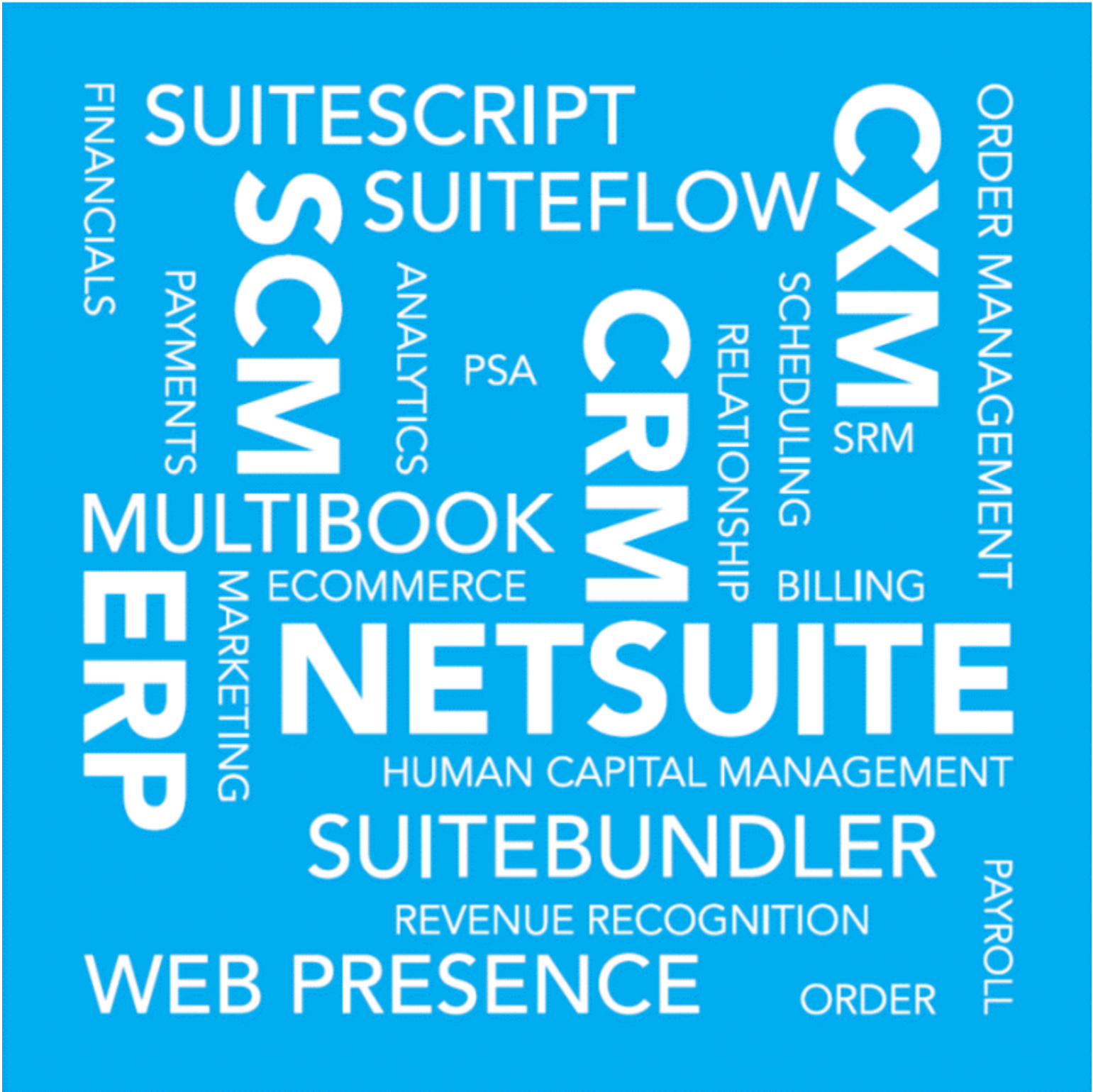


Authentication Guide



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.

Table of Contents

Authentication Overview	1
Understanding Password Requirements and Policies	3
NetSuite Password Requirements	3
PCI Compliance Password Requirements	6
Password Reset Tips for Administrators	6
NetSuite Login Pages	9
Types of Login Pages for Your NetSuite Account	9
Creating Custom Pages for Login to Your NetSuite Account	9
Customizing Login and Logout Behavior	12
NetSuite Login Pages and iFrame Prohibition	13
Enabling and Creating IP Address Rules	14
Token-based Authentication	19
Getting Started with Token-based Authentication	19
Managing TBA Tokens	23
Troubleshooting Token-based Authentication (TBA)	28
Token-based Authentication and RESTlets	37
Token-based Authentication and Web Services	37
TBA: Calling a Token Endpoint	38
Two-Factor Authentication	40
Types of Two-Factor Authentication Supported in NetSuite	40
Managing Two-Factor Authentication	42
Designating Two-Factor Authentication Roles	43
Users and Trusted Devices for Two-Factor Authentication	45
2FA by Phone or Authenticator App	46
Resetting a User's 2FA Settings	47
Supported Countries: SMS and Voice Call	48
2FA Using RSA Tokens	57
Working with RSA Tokens in Multiple Account Types	59
Uploading RSA Token Information into NetSuite	60
Associating Users with RSA Tokens	61
Searching for an RSA Token	64
Device ID Authentication	66
Device ID and the SCIS SuiteApp	66
Managing Devices on the List of devices Page	66
The Device Record	68
Creating Device Records Manually	68
Outbound Single Sign-on (SuiteSignOn)	71
SuiteSignOn Overview	71
Understanding SuiteSignOn	72
SuiteSignOn Benefits	73
SuiteSignOn Sequence Diagram and Connection Details	73
SuiteSignOn Required Features	75
Setting Up SuiteSignOn Integration	75
Creating SuiteSignOn Connection Points	76
Creating a Custom Subtab Connection Point	77
Creating a Portlet Connection Point	78
Creating a Suitelet Connection Point	79
Creating a User Event Connection Point	80
Comparing Subtab, Portlet, Suitelet and User Event Connection Points	81
Creating SuiteSignOn Records	82
Setting SuiteSignOn Basic Definitions	83
Defining SuiteSignOn Connection Points	84
Choosing User Identification Fields for SuiteSignOn	86

Using Custom Fields as SuiteSignOn User Identification	86
Dynamically Mapping User Identification Information	87
Editing SuiteSignOn Records	87
Creating a SuiteSignOn Bundle	88
Making SuiteSignOn Integrations Available to Users	89
Installing a SuiteSignOn Bundle	89
Completing Account Setup for SuiteSignOn	90
SuiteSignOn Code Samples	92
NetSuite SuiteSignOn Translation of OAuth Definitions	92
Mappings of dc and env URL Parameter Values	92
Sample SuiteSignOn HTTP Calls	93
Inbound Single Sign-on	96
Inbound Single Sign-on Overview	97
Understanding Inbound Single Sign-on	97
Setting Up Inbound Single Sign-on	100
Generating Keys Using OpenSSL	101
Creating the Initial Mapping of the Administrator Role for Inbound Single Sign-on	102
Creating Single Sign-on Code Using SSOUrl	103
SuiteTalk (Web Services) Single Sign-on Operations	110
Error Handling for Inbound Single Sign-on	111
Setting Up a Single Sign-on Only Role	111
Mapping Users and Roles for Inbound Single Sign-on Access to NetSuite	112
Technical Summary of Inbound Single Sign-on	113
SAML Single Sign-on	115
Complete Preliminary Steps in NetSuite for SAML SSO	116
Configure NetSuite with Your Identity Provider	121
Complete the SAML Setup Page	122
IdP Metadata and SAML Attributes	125
Update Identity Provider Information in NetSuite	128
Interactions with NetSuite Using SAML	129
SAML SSO in Multiple NetSuite Account Types	129
NetSuite SAML Certificate References	131
Remove SAML Access to NetSuite	131
OpenID Single Sign-on	133
Setting Up Google Account Access to NetSuite	134
Logging in to NetSuite from G Suite	137
Removing Google OpenID Access to NetSuite	138

Authentication Overview

NetSuite supports many types of authentication, both for authenticating into the NetSuite User Interface (UI) and authentication methods for integrations.

Authenticating into the NetSuite UI

Familiar to many users is authentication by user credentials, that is, supplying a username and password to login to the NetSuite UI. See the help topic [Your User Credentials](#) for information for users.

Topics for Account Administrators include [Understanding Password Requirements and Policies](#), [NetSuite Login Pages](#), and [Enabling and Creating IP Address Rules](#).

Two-Factor Authentication, or 2FA, can protect your company from unauthorized access to your data. NetSuite supports two types of 2FA, including 2FA by Phone or Authenticator App, and 2FA using RSA SecureID hardware and software tokens. For more information, see [Two-Factor Authentication](#).

Single Sign-on Overview

NetSuite supports several different types of single sign-on (SSO). SSO is a transparent authentication scheme that enables the seamless linking of applications and at the same time maintaining application-specific access control. Single sign-on eliminates the need for users to log in to each application separately.

NetSuite supports the following methods for inbound single sign-on access to the NetSuite UI:

- Inbound single sign-on using authentication from a third party identity provider compliant with SAML v2.0. See [SAML Single Sign-on](#). See also [SAML Single Sign-on Access to Web Store](#).
- Token-based inbound single sign-on when the Inbound Single Sign-on feature is enabled. See [Inbound Single Sign-on](#). See also [Inbound Single Sign-on Access to Web Store](#).
- Inbound single sign-on from Google Apps, using Google OpenID Connect authentication. See [OpenID Single Sign-on](#).

NetSuite supports outbound single sign-on when the SuiteSignOn feature is enabled. See [Outbound Single Sign-on \(SuiteSignOn\)](#). SuiteSignOn access to NetSuite from your web store is supported. See the help topic [Outbound Single Sign-on \(SuiteSignOn\) Access from Your Web Store](#).

Authentication for Integrations

NetSuite user credentials can be used for both web services integrations and for RESTlets. For more information, see the **User Credentials** section in [Authentication for Web Services](#) and [Authentication for RESTlets](#).

Inbound Single Sign-on is another authentication option for web services integrations. See the **Inbound Single Sign-on (Inbound SSO)** section in [Authentication for Web Services](#) for more information.

NetSuite also offers our own Token-based Authentication, or TBA, enabling client applications to use a token to access NetSuite through APIs. TBA eliminates the need for RESTlets and web services integrations to store user credentials. TBA allows integrations to comply with any authentication policy that is deployed in a NetSuite account for UI login, such as SAML Single Sign-on, Inbound Single Sign-on, and Two-Factor Authentication. You can use Two-Factor Authentication roles and roles with SAML

Single Sign-on permissions with TBA. For more information, see [Token-based Authentication](#). See also the **Token-based Authentication** section in [Authentication for Web Services](#).

Outbound Single Sign-on, called SuiteSignOn in NetSuite, is another authentication method supported for integrations. See [Outbound Single Sign-on \(SuiteSignOn\)](#). Only SuiteTalk (web services) is supported for outbound SSO calls.

NetSuite also offers Device ID authentication. Device ID authentication was developed for use with the SuiteCommerce In Store application, but NetSuite customers could develop their own applications to take advantage of the availability of Device ID authentication. See [Device ID Authentication](#) for more information.

Authentication Matrix

The following table shows the authentication methods supported for login to NetSuite.

	NetSuite Application	SuiteCommerce	SuiteTalk (web services)	SuiteScript RESTlets
User Credentials	Supported	Supported	Supported	Supported
Token-based Authentication			Supported	Supported
Two-Factor Authentication	Supported			
Inbound Single Sign-on	Supported	Supported	Supported	
SAML 2.0	Supported	Supported		
Google OpenID Connect	Supported			

Understanding Password Requirements and Policies


For information about password requirements and policies, see the following:

- [NetSuite Password Requirements](#)
- [PCI Compliance Password Requirements](#)
- [Password Reset Tips for Administrators](#)

NetSuite Password Requirements

For NetSuite users who log in with a non-customer center role, password validation is based on a combination of the following:

- Password settings that can be modified by account administrators at Setup > Company > General Preferences. See the following sections for more information:
 - [Password Policy](#)
 - [Minimum Password Length](#)
 - [Password Expiration in Days](#)
- PCI DSS requirements apply to users with the View Unencrypted Credit Cards permission. See [PCI Compliance Password Requirements](#).
- System requirements that cannot be modified. See [System-Defined Password Requirements](#).

 **Note:** Users are locked out for 30 minutes after six consecutive attempts to log in to NetSuite with an incorrect password.

Password Policy

Built-in password policies support three levels of password validation for NetSuite users. These policies enforce the following requirements for password length and content:

- **Strong:** minimum length of 10 characters, at least three of these four character types —uppercase letters, lowercase letters, numbers, non-alphanumeric ASCII characters
- **Medium:** minimum length of eight characters, at least two of these four character types —uppercase letters, lowercase letters, numbers, non-alphanumeric ASCII characters
- **Weak (Not Recommended):** minimum length of six characters

Note the following details about password policies:

- The selected password policy determines the minimum acceptable value for the Minimum Password Length field. The policy does not affect the Password Expiration in Days field value.
- All NetSuite accounts are set to a Strong policy by default.



Important: The Strong password policy was set as the default for each account when it was upgraded to 2014.1. By default, the Strong policy is enforced for all new users added after that upgrade. However, this policy is only enforced for users who existed before the upgrade when these users change their passwords. It is strongly recommended that you set a value for Password Expiration in Days to ensure that the Strong policy is applied to users that existed before the upgrade. (See [Password Expiration in Days](#) for more information.) You also can enable the **Require Password Change on Next Login** option on employee records. You can use CSV import to update this option on many employee records at the same time.

- It is possible to reset the password policy to Medium or Weak, but this is not recommended.



Warning: If any users in your account have the View Unencrypted Credit Cards permission, PCI password requirements take precedence. See [PCI Compliance Password Requirements](#) for more information.

- If a user has access to multiple NetSuite accounts that have different password policies, the strongest policy is enforced for that user. A user is defined as an email and password pairing.
- The password policy is not applied to users logging in to NetSuite with a customer center role and to customers who register on your website. See [Customer Roles and Passwords](#) for more information.

Minimum Password Length

The Minimum Password Length is the minimum number of characters required for user passwords. Note the following details:

- The default value for this field is determined by the selected password policy. Because the default password policy is Strong, the default Minimum Password Length is 10 characters.
- You can, if desired, make the minimum password length value longer than the minimum required by the policy. You cannot make this value shorter.
- Minimum password length for customer center roles is six characters. See [Customer Roles and Passwords](#) for more information.

Password Expiration in Days

The Password Expiration in Days is the number of days a login password can be used before a user is prompted to change it. Note the following details:

- Days are calculated from the date that each user last changed their password, not from the date that the company preference is changed.




Note: As of December 10, 2015, valid values are 1-365. Values entered before that date are not affected by this limit. However, if any data on the General Preferences page is changed, only valid values within this range will be accepted for the Password Expiration in Days field. For accounts provisioned after this date, the value for Password Expiration in Days is set to 180 days by default.

- Before 2013.2, this field was blank by default. For all new accounts as of 2013.2 or later, a value of 180 days is the default, ensuring password rotation at least every six months. NetSuite did not reset

the value of the Password Expiration in Days field for accounts that existed before that release. However, it is strongly recommended that administrators of these accounts set this value to a minimum of 180 days.

- To comply with Payment Card Industry (PCI) standards, employees with access to view unencrypted credit card numbers are automatically required to change their passwords every 90 days, unless the limit set here is shorter. See [PCI Compliance Password Requirements](#) for more information.
- Dates of the previous password change and current password expiration are displayed in the user's My login audit portlet.

 **Note:** Passwords for users with a customer center role do not expire. See [Customer Roles and Passwords](#) for more information.

System-Defined Password Requirements

The following password requirements are always enforced by the system and cannot be changed by account administrators:

- A prior password may never be reused.
- There must be a significant difference between a new password and the last password. (For example, a user cannot change a password from `MyWord!123` to `MyWord!145`.)
- Easy-to-guess passwords, such as common names, words, and strings like `abcd123456`, are prohibited.
- Non-ASCII characters are considered illegal characters and are prohibited.
- The minimum password length must be at least the minimum required by the selected password policy.
- Passwords must contain the appropriate variety of character types specified by the selected password policy:

Character types are:

- Uppercase alphabet (A, B, ... Z)
- Lowercase alphabet (a, b, ... z)
- Number (1, 2, 3, 4, 5, 6, 7, 8, 9, 0)
- Non-alphanumeric ASCII characters, for example `` ~ ! @ # $ % ^ & *) ; ' [] { }`.

Immediate Feedback on Password Changes

As they enter a new password, users receive immediate feedback on compliance with password requirements. You receive the same kind of feedback when you enter a user password on the Access subtab of an employee, partner, vendor, or customer record.

For more information about how users can change their passwords, see the help topic [Change Password Link](#).

Note: The Password Criteria fields are shown on any page where a user changes a password. It ensures that the user can tell whether the proposed password meets the security rules enforced by the system.

PCI Compliance Password Requirements

When using the Credit Card Payments feature, be aware of the Payment Card Industry Data Security Standard (PCI DSS) password requirements. Users with the View Unencrypted Credit Cards permission must change their NetSuite passwords at least every ninety (90) days.

If the number of days set in the Password Expiration in Days field on the General Preferences page is less than ninety days, that requirement remains in effect. For example, if your company is set to expire passwords every sixty days, your password expiration date does not change. However, if your company is set to expire passwords every 120 days, this setting automatically changes to 90 days for users with the View Unencrypted Credit Cards permission.

In addition, passwords for those with access to unencrypted credit card numbers must have a minimum of seven (7) characters. If the number of characters set in the Minimum Password Length field on the General Preferences field is greater, that greater requirement remains in effect.

All users with access to unencrypted credit card numbers must change passwords to comply with the PCI requirements.

Password Reset Tips for Administrators

In most cases, changing a NetSuite password is self-service. However, there are occasions when an administrator must change a user's password. This can happen, for example, when users forget the answers to their own security questions.

Employee, partner, and vendor roles are considered non-customer center roles. Customers have customer center roles. One person could use the same email address (the NetSuite username) and could be assigned both non-customer center roles and a customer center role. However, these would be treated by the system as two different users, because the information is maintained separately. Changing the password for non-customer center roles has no effect on the password of the customer center role.

See the following sections for more information.

- [Password Reset for Employees, Partners, and Vendors](#)
- [Customer Roles and Passwords](#)

Password Reset for Employees, Partners, and Vendors

There are two methods for resetting an employee, partner, or vendor password.

- Self-service password reset: On the NetSuite login page, a user can click the **Forgot Your Password?** link. The user will receive an email with a link to reset the password. The link in the email will expire after 60 minutes. See the help topic [Getting Access When You Forget Your Password](#) for information for users.
- Administrator-initiated password reset: This is similar to the initial password setup when the employee, partner, or vendor record was created. For instructions for changing a user's password, see *Working with NetSuite User Passwords*.



Important: An administrator must have access to all of the accounts to which a user has access to change that user's password.

Customer Roles and Passwords

There are two ways to create customers:

- When an administrator (or any user with the necessary permission) creates a Customer record in NetSuite and assigns a user a customer center role.
- When a visitor to your website registers an email address (a username) and creates a password. This action creates Lead record in your NetSuite account. Lead and Prospect records can be converted into Customer records.

Password Reset for Customers

There are three methods for resetting a customer's password.

- Self-service password reset:
 - If the Customer Access feature is enabled in this account, a user can click the **Forgot Your Password?** link on the NetSuite Customer Center login page. The user will receive an email with a link to reset the password. The link in the email will expire after 60 minutes. See [Types of Login Pages for Your NetSuite Account](#) and [Creating Custom Pages for Login to Your NetSuite Account](#) for more information.
 - On a website login page, customers can click the **Forgot Your Password?** (or similarly named) link. The customer will receive an email with a link to reset the password. The link in the email will expire after 60 minutes. For more information, see the help topic [Web Store Password Recovery Email Messages](#).
- Administrator-initiated password reset: This is similar to the initial password setup when the Customer record was created. For instructions, see *Working with NetSuite User Passwords*.

Password Requirements for Customers

- The minimum password length for customers is six characters. Passwords in existence for customers before 2014.1 were not affected by this change.

- NetSuite password policies and requirements (other than the requirement for a minimum password length of six characters) do not apply to customers.
- Passwords for customers do not expire. That is, the value set in the account for the Password Expiration in Days field is not applied to customer passwords.
- Setting up security questions and answers is not required for customers.

NetSuite Login Pages


See the following for more information about login pages for your NetSuite account:

- [Types of Login Pages for Your NetSuite Account](#)
- [Creating Custom Pages for Login to Your NetSuite Account](#)
- [Customizing Login and Logout Behavior](#)
- [NetSuite Login Pages and iFrame Prohibition](#)

Types of Login Pages for Your NetSuite Account

In 2016, we began a project to segregate Customer Center roles from other NetSuite roles. (If you need more information about this project, see the help topic [FAQ: Customer Center Roles Segregation](#).) One result of this project is that there are two different types of login pages for access to the NetSuite UI.

- **Standard login pages:** The standard login pages are <https://system.netsuite.com/pages/customerlogin.jsp> or <https://system.netsuite.com/pages/login.jsp>. These login pages are for users with any role except for a user logging in with a Customer Center role.
- **Customer Center login pages:** We also provide a unique login page for each NetSuite account for your users logging in with a Customer Center role. The URLs for this type of login page are in the following format: `system.<DC>.netsuite.com/app/login/secure/privatelogin.nl?c=<ACCOUNTID>`, where `<DC>` is the identifier of the data center where the account is hosted, and `<ACCOUNTID>` is the unique ID of your account. Administrators can go to Setup > Company > Setup Taks > Company Information to view the Account ID field and Customer Center Login field. The Customer Center Login field displays the unique URL for the system-provided Customer Center login page to access your account.

 **Note:** You can also create a custom login page for your users with Customer Center roles. See [Creating Custom Pages for Login to Your NetSuite Account](#) for more information.

Each type of login page displays a forgot your password link. Users can click the link, and an email is sent to the user's email address with instructions for resetting a forgotten password.

Creating Custom Pages for Login to Your NetSuite Account

NetSuite provides standard login pages for your NetSuite account. However, you can also create custom pages for login. For example, you might want to include your company's branding on the login page.

In 2016, we began the gradual process of segregating Customer Center roles from non-Customer Center roles. (See the help topic [FAQ: Customer Center Roles Segregation](#) for answers to common questions about this change.) The segregation of Customer Center roles project completed in the second quarter of 2017. This segregation of roles means that a separate login page for Customer Center roles is required. You can use the system-provided Customer Center login page for this purpose, or you can create your own custom login page, or pages.

Note: In 2017.2, Administrators can specify that their custom Customer Center login page be served instead of the default Customer Center login page. If you have a custom login page for your Customer Center, ensure it has been uploaded to your NetSuite File Cabinet. Then, go to Setup > Company > Company Preferences > General Preferences and scroll down to the Customer Center Login Page field. Select the filename for your Customer Center login page.

Your custom login page, and any images displayed on it, must be uploaded to the images folder in the file cabinet at Documents > Files > Images. Also, you must use the secure URL displayed on the file record in any tags you use to display content on your login page.

If you decide to create a custom login page (for Customer Center roles or for non-Customer Center roles, or for both types of roles) the login page must be **hosted in the NetSuite File Cabinet**. You can then display a link to the custom login page on a different page on your website.

Important: Security best practices do not allow presenting login fields to your NetSuite account in an iFrame on your web page. The following approved procedure details how to provide login access to your NetSuite account.

Creating a Custom Login Page

The following procedure describes how to create custom login pages. If you are creating a custom login page for Customer Center roles, you must know your account ID to complete this procedure. The variable in the following code example is <ACCOUNT_ID>.

To locate your account ID, go to Setup > Company > Setup Tasks > Company Information. The account ID field is located near the bottom of the right column.

To create a custom login page for your NetSuite account:

1. Create a custom login page in HTML, using the code below to display the NetSuite account login fields. Save the HTML file to your hard drive.
 - If you are creating a custom login page for non-Customer Center roles, you could, for example, name the file `NSlogin.html`. You do not have to modify the code shown below if you are creating a non-Customer Center login page.
 - If you are creating a login page for Customer Center roles, you could name the file, for example, `NSprivatelogin.html`. You must modify two lines in the sample. In each line you modify, replace the variable <ACCOUNT_ID> with your account ID.

- Modify the first line (the post action link) as shown:

```
<form method="post" action="/app/login/secure/privatelogin.nl?
c=<ACCOUNT_ID>
```

- Modify the href line for the Forgot your password link as shown:

```
<href="/app/login/preparepwdreset.nl?private=t&c=<ACCOUNT_ID>">
```

Note: The following code only represents the basic required fields for login to your NetSuite account. You can add content to this file, but you must use a secure URL to refer to any additional files.

```
<!--The post action link below is for a non-Customer Center login page-->
<form method="post" action="/app/login/nlogin.nl">
<!--For a Customer Center login page, modify the post action link as specified in step 1.-->
<table border="0" cellspacing="0" cellpadding="3">
<tr>
```

```

        <td>
            Email address:<input name="email" size="30">
        </td>
    </tr>
    <tr>
        <td>
            Password:<input name="password" size="30" type="password">
        </td>
    </tr>
    <tr>
        <td>
            <!--The href link below is for a non-Customer Center login page-->
            <a href="/app/login/preparepwdreset.nl">Forgot your password?</a>
            <!--For Customer Center login page, modify the href link as specified in step 1.-->
        </td>
    </tr>
    <tr>
        <td>
            <input type="submit" name="submitter" value="Login" >
        </td>
    </tr>
</table>
</form>

```

2. Go to the Images folder in the NetSuite File Cabinet (Documents > Files > Images).
3. Click **Add File**, and then select the appropriate HTML file for the custom login page that you created in step 1. Ensure that the "Available without login" box is selected.
4. Click **Open**. The HTML file for your custom login page is uploaded to the file cabinet. You can also add any additional files you want to use for content on your custom login page to this folder. Ensure that the "Available without login" box is selected for these files.
5. Determine the secure URL for your custom login page. You will use the secure URL later to display the link to your custom login page.
 - a. Go to the Images folder in the NetSuite File Cabinet (Documents > Files > Images).
 - b. Click **Edit** next to the HTML file for your custom login page.
 - c. Copy the NetSuite URL that starts with `https://system. ...`. You will use this URL to create a link to your login page.
6. Reference your custom login page from your website. You can now link to your custom login page from any external source by adding an `href` that uses the secure URL you copied in step 5.c.

For example:

```
<a href="https://system.netsuite.com/...>Login Here</a>
```

Do not copy the example! Use the URL you copied in step 5.c. in your `href`.



Important: The HTML file for the custom login page you created in step 1 **must** be hosted in the NetSuite File Cabinet. The external source hosting the link **does not** have to be in the NetSuite File Cabinet.

Security policies and contractual agreements prohibit displaying a NetSuite login page in an iFrame. For more information, see [NetSuite Login Pages and iFrame Prohibition](#).

Customizing Login and Logout Behavior

You can customize the behavior when a user logs in to NetSuite and the behavior when a user logs out of NetSuite.

Customizing Login Page Behavior

Using a Redirect Parameter

You can redirect users, after login, to a specific landing page in the NetSuite UI. For example, you might want to have NetSuite open up on a Customer record, or a Support Case record.

To redirect a user to a particular page after login:

1. Add a redirect hidden field to the login form in your hosted HTML page, for example:

```
<input type="hidden" name="redirect" value="/app/center/card.nl?success=true" >
```

2. Follow the steps in the procedure in the section [Creating a Custom Login Page](#).

Displaying an Error Message

When someone attempts to login with the wrong password or email, you can display an error on your hosted login page. This lets you maintain consistent company branding on the login page, instead of redirecting to a generic NetSuite error page.

To display an error message on your custom login page:

1. Add an error redirect hidden field to the login form in your hosted HTML page, for example:

```
<input type="hidden" name="errorredirect" value="/core/media/media.nl?id=572&c=TSTDVR1154923&h=b0c2553e7af5afb07ef2&success=false" >
```

2. Create a separate version of your HTML login page that includes the error message, or implement conditional logic in your custom HTML login page.
3. Follow the steps in the procedure in the section [Creating a Custom Login Page](#).

Customizing Logout Behavior

You can connect your company website's look and feel with the NetSuite application by specifying a landing page when users log out of a NetSuite center.

To specify a landing page for logout:

1. Go to Setup > Company > General Preferences
2. Click the **Centers** subtab to select the appropriate center.
3. Enter the URL for the **Sign Out Landing Page**.

NetSuite Login Pages and iFrame Prohibition

Security policies and contractual agreements prohibit displaying a NetSuite login page in an iFrame. This prohibition is documented in [Secure Login Access to Your NetSuite Account](#).

As part of a continuing commitment to provide the most secure environment possible, since January 2015, we have been enforcing the prohibition against the use of iFrames on the following login pages:

- /pages/customerlogin.jsp

For example: <https://system.netsuite.com/pages/customerlogin.jsp>

- /pages/login.jsp

For example: <https://system.netsuite.com/pages/login.jsp>

This prohibition is intended to protect against what is known as a clickjacking attack. For more information on defending against this vulnerability, see https://www.owasp.org/index.php/Clickjacking_Defense_Cheat_Sheet. This enforcement change is in accordance with best practices outlined in RFC7034 - HTTP Header Field X-Frame-Options.

To allow logins through NetSuite, you must create a login page hosted on the NetSuite secure server and display a link to this login page on a different page.

See [Creating Custom Pages for Login to Your NetSuite Account](#) for more information. See also [FAQ: Customer Center Roles Segregation](#).

Enabling and Creating IP Address Rules

You can limit access to your company's NetSuite account by entering IP address rules. Only computers with IP addresses that match those you have entered will be permitted to access your NetSuite account. For example, you may want employees logging in to your NetSuite account from a trusted location as an additional authentication requirement.

Note: To further secure the user login process, NetSuite two-factor authentication is the preferred alternative to IP address restrictions. For more information, see [Two-Factor Authentication](#).

Warning: IP addresses were designed primarily to serve host identification and addressing, thus they cannot fully serve as a reliable second factor for user authentication. Consider the following precautions, but be advised that two-factor authentication is **strongly** recommended.

- Only public IPv4 addresses can be used. Private IPv4 addresses cannot be used outside of your private network.
- IPv6 addresses are not supported.
- Make sure that you are the only owner of the public IPv4 address and that it is not shared among multiple ISP clients.

With the increasing number of network devices, it is difficult to determine the IPv4 address of the client reliably. Increased scarcity of IPv4 addresses is leading ISPs to use Carrier-Grade NAT (CGN), Large-Scale NAT (LSN), and shorter Dynamic Host Configuration Protocol (DHCP) lease times. The client IPv4 address is not usually designated to one client, nor is it static.

- Any IP packet can be spoofed and the source-address modified or crafted.
- Any IP address being rented to you cannot be treated as a reliable authentication factor.

You can restrict access at the company level or at the employee level. If you want to use IP address restrictions at the company level, check the Inherit IP Rules From Company box on employee records. Employees then will only have access to those computers you specify on the Set Up Company page. At the employee level, you can specify certain IP addresses on employee records if you want to limit an employee to a computer(s) within the company.

Inbound single sign-on access to NetSuite respects IP address restriction rules. Web services and SAML Single Sign-on also respect IP Address restriction rules.

Warning: SuiteAnalytics Connect access to NetSuite does not respect IP address restriction rules. Users may be able to access NetSuite data through SuiteAnalytics Connect from IP addresses that they cannot use to access the NetSuite application directly.

To enable the setting of IP address restrictions:

Important: Enabling the IP Address Rules feature does not retroactively apply IP address restrictions to preexisting customized roles.

1. Go to Setup > Company > Enable Features.
2. On the **Company** subtab, in the **Access** section, check the **IP Address Rules** box.
3. Click **Save**.

Note: IP address rules may prevent users from accessing Web queries of NetSuite data. For example, this issue occurs when a user with an IP address rule creates a Web query and sends it to other users who are logging in from different IP addresses.

Setting Up Company IP Address Rules

To set up IP address rules for your company:

1. Go to Setup > Company > Company Information.
2. In the **Allowed IP Addresses** field, enter valid IP addresses (in dotted decimal notation) from which you want employees in your company to access your account. Each of the numbers in the four segments (the numbers between the dots) must be between 0 and 255.

Warning: Be sure that you have entered the correct IP addresses before you log out so that you and your employees can log back in.

Use the following formats:

Important: You can enter up to 4000 characters. Use shorter forms of notation to enter addresses (such as 123.45.67.80-99 or 123.45.67.80/24 in the following examples) if necessary.

- A single IP address, such as 123.45.67.89
- A range of IP addresses, with a dash and no spaces between, such as 123.45.67.80-123.45.67.99. You can use 123.45.67.80-99 to indicate the same range.
- A list of IP address separated by spaces or commas such as 123.45.67.90, 123.45.67.97,...
- An IP address with full netmask, such as 123.45.67.80/255.255.255.0

Note: A netmask defines which bits of the IP address are valid, the example means "use the first three segments (255.255.255), but not the fourth segment (0)".

- An IP address and bitmask, such as 123.45.67.80/24

Note: The "24" indicates the number of bits from beginning to use in the validation – the same IP addresses are valid as in the previous example (255 means 8 bits).

- An IP address and mask, such as 209.209.48.32/255.255.0.0 or 209.209.48.32/16.

Warning: Think carefully when using this type of notation. The mask is a binary number. For example, the IP address and mask 12.34.56.78/12.34.56.78 does not indicate only one IP address is allowed. The IP address 140.34.56.78 matches the mask in this example. There are more IP addresses that match the mask than are immediately obvious.

- The text "NONE" – denies access from all IP addresses.
- The text "ALL" – allows all IP addresses.

3. Click **Save**.

Now, when you or other employees log in to NetSuite, if at least one rule is defined, the IP address of the computer that is being used must match the rule(s) defined. If the computer does not match the IP

address rule(s) defined, login fails and a message is displayed that login is not allowed from the current IP address.

Setting Up Individual IP Address Rules

To allow an employee to only access specific machines, you can edit the employee record and enter one IP addresses for computers that can be accessed.

Employees whose records were created before the IP Address Rules feature was enabled inherit the rules you set at Setup > Company > Company Information by default.


To set up IP address rules for individual employees:

1. Go to Lists > Employees > Employees..
2. Click **Edit** next to the employee you want set IP address rules for.
3. Click the **Access** subtab.
4. Check the **Inherit IP Rules from Company** box to give this employee access to the IP addresses defined at Setup > Company > Company Information.


Clear this box to allow access for this employee **only** at the address you enter in the **IP Address Restriction** field.

If you check this box **and** enter addresses in the IP Address Restriction field, this employee will have access to both the addresses listed at Setup > Company > Company Information and the addresses you list on this record.


5. To give this employee access to use specific machines, clear the **Inherit IP Rules from Company** box, and list the IP addresses in the **IP Address Restriction** field.

 **Note:** Enter valid IP addresses (in dotted decimal notation) from which you want this employee to access your account. Each of the numbers in the four segments (the numbers between the dots) must be between 0 and 255.


Use the following formats:

 **Important:** You can enter up to 4000 characters. Use shorter forms of notation to enter addresses (such as 123.45.67.80-99 or 123.45.67.80/24 in the following examples) if necessary.


- A single IP address, such as 123.45.67.89
- A range of IP addresses, entered with a dash and no spaces between, such as 123.45.67.80-123.45.67.99. You can use 123.45.67.80-99 to indicate the same range.
- A list of IP address separated by spaces or commas such as 123.45.67.90, 123.45.67.97,...
- An IP address with full netmask, such as 123.45.67.80/255.255.255.0

 **Note:** A netmask defines which bits of the IP address are valid, the example means "use the first three segments (255.255.255), but not the fourth segment (0)"

- An IP address and bitmask, such as 123.45.67.80/24

 **Note:** The "24" indicates the number of bits from beginning to use in the validation – the same IP addresses are valid as in the previous example (255 means 8 bits).

- An IP address and mask, such as 209.209.48.32/255.255.0.0 (allows 209.209.*.*)

 **Warning:** Think carefully when using this type of notation. The mask is a binary number. For example, the IP address and mask 12.34.56.78/12.34.56.78 does not indicate only one IP address is allowed. The IP address 140.34.56.78 matches the mask in this example. There are more IP addresses that match the mask than are immediately obvious.

- The text "NONE" – denies access from all IP addresses.
- The text "ALL" – allows all IP addresses.
- If you leave the field blank, IP address restrictions are inherited from the company level.

6. Click **Save**.

Setting Up Roles without IP Address Restrictions

You can make exceptions to your IP address rules by customizing roles. By default, all roles are restricted by the IP address rules you set at Setup > Company > Company Information and on employee records. You can customize roles, however, to create roles that are not restricted by these rules. This way, your employees can access certain roles from anywhere and restricted roles from only the machines you specify.

To customize a role so that it does not have IP address restrictions:

1. Go to Setup > Users/Roles > Manage Roles.
2. Click **Customize** next to the role type you want to assign without IP rule restrictions.
3. In the **Name** field, enter or accept the name for this non-restricted role.
4. Clear the **Restrict this role by IP Address** box.
5. On the subtabs below, click the line of any permission you want to edit.
6. Change the level of permission to **View**, **Full**, **Edit**, **None** or **Create**.
7. Click **Done**.
8. Click **Save**.

Now, when assigning roles on the **Access** subtab of employee records, you can assign this new custom role without IP address restriction. This employee can access the custom role from any computer, regardless of the IP address rules set on the employee record or at Setup > Company > Company Information.

If this employee has another role with IP address restrictions, the employee can only access that role from the addresses listed on the employee record or the addresses listed at Setup > Company > Company Information when the **Inherit IP Rules from Company** box is checked.

Reviewing User IP Address Restrictions

To see a list of all users and review a list of the IP addresses they are restricted to using for each assigned role, go to Setup > Users/Roles > View Login Restrictions.

Searching for User Login Restrictions

Users with the proper privileges can search for User Login Restrictions by user, role, and IP address.

To search for user login restrictions:

1. Go to Setup > Users/Roles > View Login Restrictions.
2. Click **Search** in the upper right corner of the page.
3. On the search page, enter your desired search parameters in the available **User**, **Role**, and **IP Addresses Allowed to Login** fields.
 - For more information on entering search parameters, see the help topic [Defining a Simple Search](#).
 - If you need help in defining filters for a simple search, see the help topic [Tips for Defining Simple Search Filters](#).
 - If you need more search criteria, check the **Use Advanced Search** box.
If you need help, see the help topic [Defining an Advanced Search](#).
4. Click **Submit**.

Token-based Authentication

NetSuite supports token-based authentication (TBA) a robust, industry standard-based mechanism that increases overall system security. This authentication mechanism enables client applications to use a token to access NetSuite through APIs, eliminating the need for RESTlets or web services integrations to store user credentials.

The TBA feature was built for integrations. Of all the inbound single sign-on features available for use in NetSuite, TBA is the only mechanism mature enough to use with SuiteTalk (web services) and RESTlets.

In your integrations, you might need to use certain functions that require an Administrator role. Two-Factor Authentication (2FA) for Administrator roles will be enforced for all existing accounts in 2018.2. In past releases, you could not use TBA tokens for integrations that required an Administrator role. The 2018.1 release introduced support for TBA for Administrators. Using an Administrator role with a TBA token ensures your integrations will continue to work after the upgrade to 2018.2. We recommend that you transition integrations that require an Administrator role to use TBA rather than user credentials.

Note: For more information about using TBA with integrations, see the help topic [Integration Management](#).

Password rotation policies in the account do not apply to tokens, making password management unnecessary for your RESTlet and web services integrations. Token-based authentication allows integrations to comply with any authentication policy that is deployed in a NetSuite account for UI login, such as SAML Single Sign-on, Inbound Single Sign-on, and Two-Factor Authentication. You can use Two-Factor Authentication (2FA) roles and roles with SAML Single Sign-on permissions with TBA.

Getting Started with Token-based Authentication

To set up token-based authentication (TBA) in your NetSuite account, do the following:

- Enable the feature. See [Enabling the Token-based Authentication Feature](#).
- Set up roles. See [Setting Up Token-based Authentication Roles](#).
 - See also [Token-based Authentication Permissions](#).
- Assign roles to users. See [Assigning Users to Token-based Authentication Roles](#).
- Set up applications for token-based authentication. See [Creating Applications for Token-based Authentication](#).

Note: Tokens created in your production account are not copied to your sandbox during a refresh. To test token-based authentication in your sandbox, you must create tokens in the sandbox account. Each time your sandbox is refreshed, you will need to create new tokens in the sandbox.

Enabling the Token-based Authentication Feature

Before you can begin using TBA in your account, you must enable the feature.

To enable the token-based authentication feature:

1. Go to Setup > Company > Enable Features.
2. Click the **SuiteCloud** subtab.

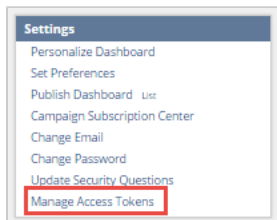
3. Scroll down to the SuiteScript section, and check the following boxes:
 - **Client SuiteScript.**
 - **Server SuiteScript.** Click **I Agree** on the SuiteCloud Terms of Service page.

Note: Enabling both the Client SuiteScript and Server SuiteScript features is required to use RESTlets with token-based authentication.

4. Scroll down to the **Manage Authentication** section, and check the **Token-based Authentication** box. Click **I Agree** on the SuiteCloud Terms of Service page.
5. Click **Save**.

After the TBA feature is enabled:

- The **Manage Access Tokens** link becomes available in the Settings portlet for users with Administrator role, or users with a role that has been assigned the appropriate permission. See the help topic [Finding Your Settings Portlet](#) if you need help locating it.



- Administrators (or users assigned the Full level of the Setup Type Integration Application permission) can create applications for use with token-based authentication. See [Creating Applications for Token-based Authentication](#). For more detailed information, see the help topic [Creating an Integration Record](#).

Setting Up Token-based Authentication Roles

Important: In past releases, TBA could not be used with the Administrator role. In 2018.1, this restriction has been removed. You can use TBA with those integrations that require the Administrator role.

If desired, an administrator can modify existing roles to add token-based authentication permissions, then assign users to those roles as needed. If you need more information about creating or customizing roles, see:

- [NetSuite Users & Roles](#)
- [NetSuite Roles Overview](#)

Token-based Authentication Permissions

Important: In past releases, TBA could not be used with the Administrator role. In 2018.1, this restriction has been removed. It is not necessary to assign token-based authentication permissions to an administrator. As of 2018.1, administrators can create access tokens for their own use. An administrator cannot create a token for another administrator.

The following token-based authentication permissions can be added to non-administrator roles as appropriate.

■ Access Token Management

For **non-administrator** roles with this permission, users:

- Can, through the NetSuite UI, create and revoke access tokens for some users with a TBA-enabled role. A user who does not have an Administrator role cannot create tokens for an Administrator.
- **Cannot** create access tokens for their own use.
- **Cannot** use access tokens to log in through RESTlets or SuiteTalk (web services).

■ User Access Tokens

Users with this permission:

- Can, through the **Manage Access Tokens** link in the Settings portlet, or by calling the token endpoint, create and revoke access tokens for their own use.
- Can use access tokens to log in through RESTlets or SuiteTalk (web services).

■ Log in using Access Tokens

Users with this permission:

- Can use access tokens to log in through RESTlets or SuiteTalk (web services).
- **Cannot** create their own access tokens through a link in the Settings portlet, or by calling the token endpoint.

Assigning Users to Token-based Authentication Roles



Important: In past releases, TBA could not be used with the Administrator role. In 2018.1, this restriction has been removed. It is not necessary to assign a modified role with token-based authentication permissions to an administrator.

After modifying roles with the appropriate token-based authentication permissions, an account administrator can assign users to those roles. The following is a brief procedure for assigning a role to an existing employee. If you need more information on assigning users to roles, see the help topic [NetSuite Users Overview](#).

To assign an employee to a token-based authentication role:

1. Go to List > Employees > Employees.
2. Click **Edit** next to the name of the employee you want to assign the token-based authentication role.
3. Click the **Access** subtab.
4. In the **Role** field, select the token-based authentication role for this employee.
5. Click **Add**.
6. Click **Save**.

Creating Applications for Token-based Authentication

Applications must be created with the Integration record for use with token-based authentication before tokens can be created and assigned to users. Administrators or users assigned the Full level of


the Setup Type Integration Application permission can create applications for use with token-based authentication.

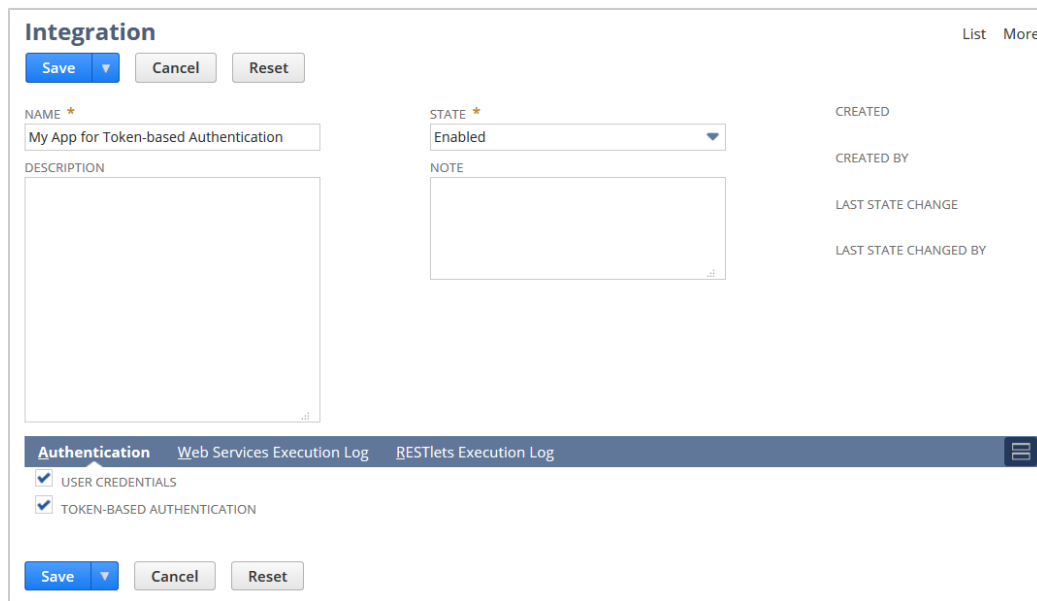
- For more information about the Integration record, see the help topic [Integration Management](#).
- For more information about using token-based authentication with web services, see the help topic [Token-Based Authentication Details](#).

To create an application using the Integration record:

The following procedure briefly describes completing an Integration record. For more detailed information about the fields in this record, see the help topic [Creating an Integration Record](#).

1. Go to Setup > Integration > Manage Integrations > New
2. Enter a **Name** for your application.
3. Enter a **Description**, if desired.
4. The application **State** is Enabled by default. (The other option available for selection is Blocked.)
5. Enter a **Note**, if desired.
6. Check the **Token-based Authentication** box on the **Authentication** subtab.

 **Note:** Clear the **User Credentials** box if you want this application to only be able to authenticate using tokens.



Integration List More

Save **Cancel** **Reset**

NAME *

STATE *

DESCRIPTION

NOTE

CREATED

CREATED BY

LAST STATE CHANGE

LAST STATE CHANGED BY

Authentication Web Services Execution Log RESTlets Execution Log

☒ USER CREDENTIALS

☒ TOKEN-BASED AUTHENTICATION

Save **Cancel** **Reset**

7. Click **Save**.

The confirmation page displays the Consumer Key and Consumer Secret for this application.

Warning: For security reasons, the only time the Consumer Key and Consumer Secret values are displayed is on the confirmation page. After you leave this page, these values cannot be retrieved from the system. If you lose or forget these credentials, you will need to reset them to obtain new values. Treat these values as you would a password. Never share these credentials with unauthorized individuals and never send them by email.

Confirmation
 Integration successfully Saved

Integration List

Edit Back Actions

APPLICATION ID F2AFB2D0-95A9-4EA8-B13C-CEDC5C13F644	STATE Enabled	CREATED 2016-04-15
NAME My App for Token-based Authentication	NOTE	CREATED BY Smith, John
DESCRIPTION		LAST STATE CHANGE 2016-04-15
		LAST STATE CHANGED BY Smith, John

Authentication Web Services Execution Log RESTlets Execution Log System Notes

☒ USER CREDENTIALS
☒ TOKEN-BASED AUTHENTICATION

Consumer key / secret

Warning: For security reasons, this is the only time that the Consumer Key and Consumer Secret values are displayed. After you leave this page, they cannot be retrieved from the system. If you lose or forget these credentials, you will need to reset them to obtain new values. Treat the values for Consumer Key and Consumer Secret as you would a password. Never share these credentials with unauthorized individuals and never send them by email.

CONSUMER KEY
641e2a605843187260c8200ebbe8e5a59b8abba08c82a3431a2638254335d685

CONSUMER SECRET
ed7ecbd4f81e9c068004d7d8126e6d95417254423b6015a873c26252ea996200

Edit Back Actions

8. Click **List** to view all Integrations in your account.

Integrations				Set Preferences
<a>New <a>Refresh				
<input type="checkbox"/> SHOW INACTIVES				
NAME	APPLICATION ID	STATE	CREATED ON	
Default Web Services Integrations		Enabled		
My App for Token-based Authentication	E3522C10-59FA-4D10-9AA1-7C3FC86D34AA	Enabled	2015-09-16 07:54:00	

After these basic setup tasks are complete, you can begin using token-based authentication in your account.

Managing TBA Tokens

Managing TBA tokens in your account includes the following:

■ Creating Tokens

There are various methods for creating tokens. In the NetSuite User Interface, the method employed depends on the permission assigned to the role. Users can also create tokens without logging in to the NetSuite UI. For more information, see the following topics:

- Access Token Management – Create and Assign a TBA Token
- User Access Token – Create a TBA Token

Note: Tokens can also be created by clicking **New Access Token** on the Access Tokens list view page. See [Viewing, Editing, Creating, and Revoking TBA Tokens](#).

- TBA: Calling a Token Endpoint
- **Viewing, Editing, and Revoking Tokens** See [Viewing, Editing, Creating, and Revoking TBA Tokens](#)
- **Search for tokens** in your account. See [Searching for TBA Tokens](#)

Access Token Management – Create and Assign a TBA Token

Important: In past releases, TBA could not be used with the Administrator role. In 2018.1, this restriction has been removed. As of 2018.1, administrators can create access tokens for their own use. An administrator cannot create a token for another administrator. Tokens for Administrators can only be created by clicking the Manage Access Tokens link in the Settings portlet or by using the token endpoint.

A user who does not have an Administrator role cannot create tokens for an Administrator. Users assigned a customized role that has the **Access Token Management** permission can create, assign, and manage a token for other users (except tokens for an Administrator role) in the company. For example, they could assign a token to those users who are assigned a role with only the **Log in using Access Tokens** permission.

Note: Tokens created in your production account are not copied to your sandbox during a refresh. To test token-based authentication in your sandbox, you must create tokens in the sandbox account. Each time your sandbox is refreshed, you will need to create new tokens in the sandbox.

To create and assign a TBA token:

1. Log in as a user with the **Access Token Management** permission.
2. Go to Setup > Users/Roles > Access Tokens > New.
3. On the Access Tokens page, click **New Access Token**.

The Access token page displays.

4. On the Access Token page:
 - a. Select the **Application Name**.
 - b. Select the **User**.

- c. Select the **Role**.
 - d. The **Token Name** is already populated by default with a concatenation of Application Name, User, and Role. Enter your own name for this token, if desired.
5. Click **Save**.

The confirmation page displays the Token ID and Token Secret.

Warning: For security reasons, the only time the Token ID and Token Secret values are displayed is on the confirmation page. After you leave this page, these values cannot be retrieved from the system. If you lose or forget these credentials, you will need to create a new token and obtain new values.

Treat these values as you would a password. Never share these credentials with unauthorized individuals and never send them by email.

User Access Token – Create a TBA Token

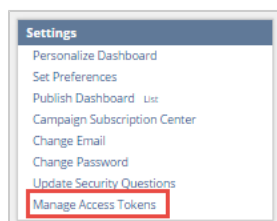
Important: In past releases, TBA could not be used with the Administrator role. In 2018.1, this restriction has been removed. As of 2018.1, administrators can create access tokens for their own use. An administrator cannot create a token for another administrator. Tokens for Administrators can only be created by clicking the Manage Access Tokens link in the Settings portlet or by using the token endpoint.

Users assigned a role that has the **User Access Token** permission can create, assign, and manage tokens for the current user and current role.

Note: Tokens created in your production account are not copied to your sandbox during a refresh. To test token-based authentication in your sandbox, you must create tokens in the sandbox account. Each time your sandbox is refreshed, you will need to create new tokens in the sandbox.

To create a token using the Manage Access Tokens link:

1. Log in using a role with the **User Access Token** permission.
2. In the **Settings** portlet, click **Manage Access Tokens**.



The My Access Tokens page displays, listing all the tokens for the current user in the current role.

My Access Tokens

List

Search

VIEW

My Access Tokens

Customise View

New My Access Token

FILTERS

REVOKED

Yes

STYLE

Normal

TOTAL: 1

EDIT VIEW	TOKEN NAME	CREATED BY	ROLE	APPLICATION	INACTIVE	CREATED
Edit View	NameChange- NetSuite Canada, Custom CEO 2 -TBA	NetSuite Canada	Custom CEO 2 -TBA	OutlookAppforTBA	Yes	2/25/2015 6:46 AM

3. Click **New My Access Token**.

The Access Token page displays.

4. On the Access Token page:
 - a. Select the **Application Name**.
 - b. The **Token Name** is already populated by default with a concatenation of Application Name, User, and Role. Enter your own name for this token, if desired.
5. Click **Save**.

The confirmation page displays the Token ID and Token Secret.

Warning: For security reasons, the only time the Token ID and Token Secret values are displayed is on the confirmation page. After you leave this page, these values cannot be retrieved from the system. If you lose or forget these credentials, you will need to create a new token and obtain new values.

Treat these values as you would a password. Never share these credentials with unauthorized individuals and never send them by email.

Viewing, Editing, Creating, and Revoking TBA Tokens

You can see a list view of tokens in your system.

To view tokens:

1. Go to Setup > Users/Roles > Access Tokens > New or go to Setup > Other Setup > Access Tokens.

The Access Tokens page displays.

EDIT VIEW	TOKEN NAME	USER	ROLE	APPLICATION	INACTIVE	CREATED	NAME
Edit View	MyOwnAppTBA - MyToken	NetSuite Canada	Custom CEO 2 - TBA	OutlookAppforTBA	No	2/26/2015 3:40 PM	NetSuite Canada
Edit View	NewAppforTBA - NetSuite Custom CEO 2 - TBA	NetSuite Canada	Custom CEO 2 - TBA	OutlookAppforTBA	No	2/25/2015 8:03 AM	NetSuite Canada
Edit View	OutlookAppforTBA - CEO-TBA	JaneTest CEO-TBA	Custom CEO 2 - TBA	OutlookAppforTBA	No	2/25/2015 6:42 AM	JaneTest CEO-TBA

2. Actions you can take from this page include:
 - Click **View** to open the Access Token page and review the details of a specific token.
 - Click **New Access Token** to open the Access Token page and create a new token. For more information, see [Access Token Management – Create and Assign a TBA Token](#).
 - Click **Edit** to open the Access Token page and:
 - Edit specific details about the token, or

- Click **Revoke** to revoke the token. For more information, see [Revoking TBA Tokens](#).
- Open the **Filters** panel to select a value for **Revoked** status (All, Yes, or No).
- Click **Search** at the top right corner of the Access Tokens page. For more information, see [Searching for TBA Tokens](#).

Revoking TBA Tokens

Revoking a token makes it inactive forever, but does not remove the token from the system. The token is still accessible for auditing purposes.

Revoke and Inactive Statuses

- When a token is revoked, it cannot be edited, and will display with an Inactive status in list views.
- When the **Inactive** box is checked for a token, the token will display as Inactive in list views, but the token can still be edited. To make the token active again, click **Edit**, clear the **Inactive** box, and click **Save**.

The screenshot shows the 'Access Token' configuration page in NetSuite. At the top, there are buttons for 'Save', 'Cancel', 'Reset', and 'Revoke'. The 'Revoke' button is highlighted with a red rectangle. Below these buttons is the 'Primary Information' section. It contains fields for 'APPLICATION NAME' (OutlookAppforTBA), 'USER' (NetSuite Canada), 'ROLE' (Custom CEO 2 -TBA), and 'TOKEN NAME *' (OutlookAppTBA_CEOToken). At the bottom of this section, there is a checkbox labeled 'InActive' which is checked, and it is also highlighted with a red rectangle. Below the checkbox, it says 'Token Id / secret'.

Additional Situations Under Which Tokens are Revoked

- When an application used for token-based authentication is deleted, all tokens associated with that application are revoked.
- When an administrator removes roles from an entity (an employee, a vendor, a partner, a customer, or a contact) the tokens are still active in the system. These active tokens cannot be used by the entity for log in to NetSuite (unless the administrator adds the roles back to the entity).
- When an administrator deletes an entity, (an employee, a vendor, a partner, a customer, or a contact) the associated tokens are deleted.

Searching for TBA Tokens

There are two methods of opening the Access Token Search page. One method is to click the Search link on the top right corner of a page. See the following procedure for the other method of opening the search page.

To search for a token:

1. Go to Setup > Users/Roles > Access Tokens > Search.

The Access Token Search page displays.

2. Enter or select from the available criteria, as appropriate.
3. click **Submit**.

For information on NetSuite's search capabilities, see:

- [Running Searches](#)
- [Saved Searches](#)

Troubleshooting Token-based Authentication (TBA)

See the following sections for TBA troubleshooting information:

- [Track TBA Tokens and Users](#)
- [TBA Related Messages in the Login Audit Trail](#)
- [Generate a Signature](#)
- [Create the Authorization Header](#)

Track TBA Tokens and Users

You can use the Login Audit Trail to track TBA tokens and users.

To track tokens and users:

1. Go to Setup > Users/Roles > User Management > View Login Audit Trail.
2. Check the **Use Advanced Search** box.
3. Click the **Results** subtab.
4. Add the following fields: **Detail**, **Token-based Access Token Name**, and **Token-based Application Name**.
5. Click **Submit**.

The **Detail** column displays error messages for any token-based authentication logins with a status of Failure.

For more information about defining Login Audit Trail searches, see the help topic [Login Audit Trail Overview](#).

TBA Related Messages in the Login Audit Trail

A good place to start troubleshooting TBA problems is the Detail column of the Login Audit Trail Results. RESTlets and web services have slightly different messages, but the meaning is similar. See the following table for information.

RESTlet	Web Service	Problem	Resolution
consumer_key_refused	consumer_key_refused	The application is in Blocked state on the integration record.	<p>Enable the application on the integration record.</p> <p>To enable the app:</p> <ol style="list-style-type: none"> 1. Go to Setup > Integration > Manage Integrations > New. 2. Select the appropriate integration record, and click Edit. 3. In the State field, change Blocked to Enabled. 4. Save the record.
consumer_key_unknown	consumer_key_unknown	The appropriate integration record could not be found.	<ul style="list-style-type: none"> ■ Ensure the consumer key is correct. ■ Ensure that the integration application is installed in the correct domain (for example: production or sandbox). ■ If this error occurs with a currently enabled application, you can attempt resetting the credentials (obtain new credentials). Important: This action might break other integrations using this application. You must update the credentials in all integrations where they are used. ■ If there is no existing integration record for this application, create one. See Creating Applications for Token-based Authentication. For more detailed information, see the help topic Creating an Integration Record.

RESTlet	Web Service	Problem	Resolution
FeatureDisabled	FeatureDisabled	The Token-based Authentication feature in NetSuite is not enabled.	Enable the feature. See Enabling the Token-based Authentication Feature .
nonce_rejected		The nonce was not long enough.	Nonce must be at least six characters long. A nonce length of 20 characters is recommended.
nonce_used	NonceUsed	The combination of nonce and timestamp has already been used by this user.	<ul style="list-style-type: none"> Ensure you generate a unique nonce for every request. Do not send the same request more than one time. If you must perform the same operation, you must generate a new TBA header for each subsequent request.
parameter_rejected		The parameter was either: <ul style="list-style-type: none"> sent twice. sent with a malformed value. sent with an empty value. 	Ensure that you: <ul style="list-style-type: none"> Only send OAuth parameters a single time. Send all values in the correct format. Do not send a parameter without a value.
permission_denied	permission_denied	The entity or role is not usable.	This error can have many reasons. <ul style="list-style-type: none"> Verify that the entity and role are both active in NetSuite. Verify the entity has access. Verify that the role has TBA permissions. Verify that the user has not made the role inactive on the user's View Role page.
signature_invalid	Invalid Signature	The request was not signed correctly.	See Generate a Signature for the correct method of signing a request.
signature_method_rejected	Unknown Algorithm	The algorithm used to create signature is not supported.	The only supported algorithms are: <ul style="list-style-type: none"> HMAC-SHA256 HMAC-SHA1
temporary_locked	temporary_locked	The user is locked out of NetSuite.	The user was locked out of NetSuite after six failed login attempts. The user must wait 30 minutes to unlock access to your account. Or, the user can ask their Administrator or system administrator for assistance.

RESTlet	Web Service	Problem	Resolution
			strator for a password reset. See Password Reset Tips for Administrators .
timestamp_refused	InvalidTimestamp	The timestamp of the request must be within plus or minus five (+ or - 5) minutes of the server time.	Ensure that: <ul style="list-style-type: none"> Your computer clocks are synchronized using the NTP protocol. Requests are sent soon after generating the TBA header. Requests are not being queued before being sent to NetSuite.
token_rejected	token_rejected	The token could not be found.	Ensure that the token: <ul style="list-style-type: none"> Is correct. Is active. Is a token for the correct domain (for example, for production or sandbox). Is a token for the correct integration application. If a token does not exist, create one. See Managing TBA Tokens .

Generate a Signature

Some users have difficulty constructing a valid signature.

The following sections describes how to correctly create a signature and provides PHP examples for each step.

- Input Parameters for the Example
- Step 1: Construct a Base String for the Signature
- Step 2: Signature Key
- Step 3: Signature

Note: All encoding in TBA is percent encoding. For more information about percent encoding, go to (<https://tools.ietf.org/html/rfc5849#section-3.6>). The examples in this section use `php rawurlencode`.

Input Parameters for the Example

These are the input parameters used for this example.

```
$url = 'https://rest.netsuite.com/app/site/hosting/restlet.nl?script=6&deploy=1&customParam=someValue&testParam=someOtherValue';
//or https://webservices.netsuite.com/services/NetSuitePort_2015_2 for webservices
$httpMethod = 'POST';
```

```

$tokenKey = '2b0ce516420110bcbd36b69e99196d1b7f6de3c6234c5afb799b73d87569f5cc';
$tokenSecret = 'c29a677df7d5439a458c063654187e3d678d73aca8e3c9d8bea1478a3eb0d295';
$consumerKey = 'ef40afdd8abaac111b13825dd5e5e2dddb44f86d5a0dd6dcf38c20aae6b67e4';
$consumerSecret = 'd26ad321a4b2f23b0741c8d38392ce01c3e23e109df6c96eac6d099e9ab9e8b5';
$signatureMethod = 'HMAC-SHA256'; //or HMAC-SHA1
$nonce = 'fjaLirsIcCGVZWzBX0pg'; //substr(str_shuffle("0123456789abcdefghijklmnopqrstuvwxyz
zABCDEFGHIJKLMNOPQRSTUVWXYZ"), 0, 20);
$timestamp = '1508242306'; //time();
$version = '1.0';
$realm = '123456'; //scompid

```

Step 1: Construct a Base String for the Signature

The first step in creating signature is constructing a Base String. This is the only step in generating a signature which is different for web services and RESTlets.

Web Services

```

$baseString = rawurlencode($realm) . "&". rawurlencode($consumerKey) . "&". rawurlencode($tokenKey) . "&". rawurlencode($nonce) . "&". rawurlencode($timestamp);

```

Example 1. Web Services Base String Example

For web services, the creation of the Base String creation is straightforward. Use percent encoding. Parameters include: realm (accountID, also called scompid), consumer key, token key, nonce, and timestamp, with the ampersand character (&) as the delimiter.

```

3829855ef40afdd8abaac111b13825dd5e5e2dddb44f86d5a0dd6dcf38c20aae6b67e4&2b0ce516420110bcbd36b69e99196d1b7f6de3c6234c5afb799b73d87569f5cc&fjaLirsIcCGVZWzBX0pg&1508242306

```

RESTlets

```

$baseString = oauth_get_sbs($httpMethod, $url, array('oauth_consumer_key' => $consumerKey,
                                                    'oauth_nonce' => $nonce,
                                                    'oauth_signature_method' => $signatureMethod,
                                                    'oauth_timestamp' => $timestamp,
                                                    'oauth_token' => $tokenKey,
                                                    'oauth_version' => $version));

```

Example 2. RESTlets Base String Example

This RESTlets example uses the oauth library. For more information, see <https://tools.ietf.org/html/rfc5849#section-3.4.1>.

```

POST&https%3A%2F%2Frest.netsuite.com%2Fapp%2Fsite%2Fhosting%2Frestlet.nl&customParam%3DsomeValue%26deploy%3D1%26oauth_consumer_key%3Def40afdd8abaac111b13825dd5e5e2dddb44f86d5a0dd6dcf38c20aae6b67e4%26oauth_nonce%3DfjaLirsIcCGVZWzBX0pg%26oauth_signature_method%3DHMAC-SHA256%26oauth_timestamp%3D1508242306%26oauth_token%3D2b0ce516420110bcbd36b69e99196d1b7f6de3c6234c5afb799b73d87569f5cc%26oauth_version%3D1.0%26script%3D6%26testParam%3DsomeOtherValue

```

Step 2: Signature Key

The signature key is used to sign the base string in the HMAC-SHA algorithm. The key is constructed from the URL-encoded values for consumer secret and token secret, with the ampersand character (&) as the delimiter.

```
$key = rawurlencode($consumerSecret) . '&' . rawurlencode($tokenSecret);
```

Step 3: Signature

The `signature` is a base64 value of the HMAC-SHA, where the message is Base String and `key` is the key from the previous step.

```
$signature = base64_encode(hash_hmac('sha256', $baseString, $key, true)); //or sha1
```

Example 3. Web Services Signature


```
76wQrUWF8i3BwfAjrNnTxjFo+Ixj9YzYgsj+HVeGQyY=
```

Example 4. RESTlets Signature

```
7mpNx1RdQn4VLSyeEwCK7jFBjGQ0blzwDSMU9Kg5Rmg=
```

Create the Authorization Header

To create the authorization header, place the correct parameter in the right place.

 **Note:** For RESTlets, each parameter must be rawurlencoded.

See the following sections:

- [Web Services Header](#)
- [RESTlet Header](#)
- [Create the RESTlet Base String Manually](#)

Web Services Header

```
$passport = "    <ns:tokenPassport soap:actor=\"http://schemas.xmlsoap.org/soap/actor/next\" s
oap:mustUnderstand=\"0\" xmlns:ns=\"urn:messages_2015_2.platform.webservices.netsuite.com\">\n"

    ."    <ns:account>". $realm .:</ns:account>\n"
    ."    <ns:consumerKey>". $consumerKey .:</ns:consumerKey>\n"
    ."    <ns:token>". $tokenKey .:</ns:token>\n"
    ."    <ns:nonce>". $nonce .:</ns:nonce>\n"
    ."    <ns:timestamp>". $timestamp .:</ns:timestamp>\n"
    ."    <ns:signature algorithm=\"\". $signatureMethod .\">\". $signature .:</ns:s
ignature>\n"
    ."    </ns:tokenPassport>;
```

Example 5. Web Services Token Passport Example

```
<ns:tokenPassport soap:actor="http://schemas.xmlsoap.org/soap/actor/next soap:mustUnderstand="0
" xmlns:ns="urn:messages_2015_2.platform.webservices.netsuite.com"
  <ns:account>123456</ns:account>
  <ns:consumerKey>f40afdd8abaac11b13825dd5e5e2dddb44f86d5a0dd6dcf38c20aae6b67e4</ns:consumerKey>
  <ns:token>2b0ce516420110bcbd36b69e99196d1b7f6de3c6234c5afb799b73d87569f5cc</ns:token>
  <ns:nonce>fjaLirsIcCGVZWzBX0pg</ns:nonce>
  <ns:timestamp>1508242306</ns:timestamp>\
  <ns:signature algorithm="HMAC-SHA256">76wQrUWF8i3BwfAjrNnTxjFo+Ixj9YzYgsj+HVeGQyY=</ns:signature>
</ns:tokenPassport>
```

RESTlet Header

```
$header = 'Authorization: OAuth '
    . 'realm="' . rawurlencode($realm) . '", '
    . 'oauth_consumer_key="' . rawurlencode($consumerKey) . '", '
    . 'oauth_token="' . rawurlencode($tokenKey) . '", '
    . 'oauth_nonce="' . rawurlencode($nonce) . '", '
    . 'oauth_timestamp="' . rawurlencode($timestamp) . '", '
    . 'oauth_signature_method="' . rawurlencode($signatureMethod) . '", '
    . 'oauth_version="' . rawurlencode($version) . '", '
    . 'oauth_signature="' . rawurlencode($signature) . '');
```

Example 6. RESTlet Header Example

```
Authorization: OAuth realm="123456", oauth_consumer_key="ef40afdd8abaac11b13825dd5e5e2dddb44f86d5a0dd6dcf38c20aae6b67e4", oauth_token="2b0ce516420110bcbd36b69e99196d1b7f6de3c6234c5afb799b73d87569f5cc", oauth_nonce="fjaLirsIcCGVZWzBX0pg", oauth_timestamp="1508242306", oauth_signature_method="HMAC-SHA256", oauth_version="1.0", oauth_signature="7mpNx1RdQn4VLSyeEwCK7jFBjGQ0blzwDSMU9Kg5Rmg%3D"
```

Create the RESTlet Base String Manually

In the following example, the Base String consists of three parts. Each step contains an image of a piece of the code to show the line numbers. To view the entire code example (without line numbers) see the following section: [The restletBaseString Function](#).

Note: POST parameters are used only with content type "application/x-www-form-urlencoded". However, this content type is not allowed by RESTlets.

1. HTTP method - line 3

Note: The HTTP method must be in uppercase.

```
1 function restletBaseString($httpMethod, $url, $consumerKey, $tokenKey, $nonce, $timestamp, $version, $signatureMethod, $postParams){
2     //http method must be upper case
3     $baseString = strtoupper($httpMethod) . '&';
4 }
```

2. Url - lines 6-16
 - Url is taken without parameters. (lines 6-12)

- Schema (http, https) and hostname must be in lowercase. (lines 13-15)

```

5 //include url without parameters, schema and hostname must be lower case
6 if (strpos($url, '?')){
7     $baseUrl = substr($url, 0, strpos($url, '?'));
8     $getParams = substr($url, strpos($url, '?') + 1);
9 } else {
10     $baseUrl = $url;
11     $getParams = "";
12 }
13 $hostname = strtolower(substr($baseUrl, 0, strpos($baseUrl, '/', 10)));
14 $path = substr($baseUrl, strpos($baseUrl, '/', 10));
15 $baseUrl = $hostname . $path;
16 $baseString .= rawurlencode($baseUrl) . '&';

```

3. Parameters - lines 19-51

- Put all OAuth, GET, and POST parameters into the array of arrays. (lines 19-37)
- Parameter names and values are urldecoded before entering into array (lines 30-34)
- The array is alphabetically sorted by parameter name. (line 40)
- The string containing all parameters is created. Each name and value is separated by the equal character (=) and each pair is separated by the ampersand character (&). Both name and value are rawurlencoded. (lines 42-50)
- The whole string containing parameters is rawurlencoded before joining with rest of the Base String (line 51)

```

19 $params = array();
20 $params['oauth_consumer_key'] = array($consumerKey);
21 $params['oauth_token'] = array($tokenKey);
22 $params['oauth_nonce'] = array($nonce);
23 $params['oauth_timestamp'] = array($timestamp);
24 $params['oauth_signature_method'] = array($signatureMethod);
25 $params['oauth_version'] = array($version);
26
27 foreach (explode('&', $getParams . "&". $postParams) as $param) {
28     $parsed = explode('=', $param);
29     if ($parsed[0] != "") {
30         $value = isset($parsed[1]) ? urldecode($parsed[1]): "";
31         if (isset($params[urldecode($parsed[0])])) {
32             array_push($params[urldecode($parsed[0])], $value);
33         } else {
34             $params[urldecode($parsed[0])] = array($value);
35         }
36     }
37 }
38
39 //all parameters must be alphabetically sorted
40 ksort($params);
41
42 $paramString = "";
43 foreach ($params as $key => $valueArray){
44     //all values must be alphabetically sorted
45     sort($valueArray);
46     foreach ($valueArray as $value){
47         $paramString .= rawurlencode($key) . '=' . rawurlencode($value) . '&';
48     }
49 }
50 $paramString = substr($paramString, 0, -1);
51 $baseString .= rawurlencode($paramString);
52 return $baseString;
53 }

```


The restletBaseString Function

```
function restletBaseString($httpMethod, $url, $consumerKey, $tokenKey, $nonce, $timestamp, $version, $signatureMethod, $postParams){
    //http method must be upper case
    $baseString = strtoupper($httpMethod) . '&';

    //include url without parameters, schema and hostname must be lower case
    if (strpos($url, '?')){
        $baseUrl = substr($url, 0, strpos($url, '?'));
        $getParams = substr($url, strpos($url, '?') + 1);
    } else {
        $baseUrl = $url;
        $getParams = "";
    }
    $hostname = strtolower(substr($baseUrl, 0, strpos($baseUrl, '/', 10)));
    $path = substr($baseUrl, strpos($baseUrl, '/', 10));
    $baseUrl = $hostname . $path;
    $baseString .= rawurlencode($baseUrl) . '&';

    //all oauth and get params. First they are decoded, next alphabetically sorted, next each key
    and values is encoded and finally whole parameters are encoded
    $params = array();
    $params['oauth_consumer_key'] = array($consumerKey);
    $params['oauth_token'] = array($tokenKey);
    $params['oauth_nonce'] = array($nonce);
    $params['oauth_timestamp'] = array($timestamp);
    $params['oauth_signature_method'] = array($signatureMethod);
    $params['oauth_version'] = array($version);

    foreach (explode('&', $getParams."&". $postParams) as $param) {
        $parsed = explode('=', $param);
        if ($parsed[0] != "") {
            $value = isset($parsed[1]) ? urldecode($parsed[1]): "";
            if (isset($params[urldecode($parsed[0])])) {
                array_push($params[urldecode($parsed[0])], $value);
            } else {
                $params[urldecode($parsed[0])] = array($value);
            }
        }
    }

    //all parameters must be alphabetically sorted
    ksort($params);

    $paramString = "";
    foreach ($params as $key => $valueArray){
        //all values must be alphabetically sorted
        sort($valueArray);
        foreach ($valueArray as $value){
            $paramString .= rawurlencode($key) . '=' . rawurlencode($value) . '&';
        }
    }
    $paramString = substr($paramString, 0, -1);
    $baseString .= rawurlencode($paramString);
}
```

```
return $baseString;
}
```

Token-based Authentication and RESTlets

Important: In past releases, TBA could not be used with the Administrator role. In 2018.1, this restriction has been removed. As of 2018.1, administrators can create access tokens for their own use. An administrator cannot create a token for another administrator.

The following details about using token-based authentication with RESTlets (TBA with RESTlets) are provided here for your convenience. For more information, see the help topic [Authentication for RESTlets](#).

Note: Web Services Only roles are only for access to NetSuite through web services. Roles with the Web Services Only restriction will not work with RESTlets.

For more information and examples, see the following topics:

- [Authentication for RESTlets](#), especially:
 - [Setting up Token-based Authentication for a RESTlet integration](#)
 - [Using NLAAuth in the Authorization Header](#)
- [TBA: Calling a Token Endpoint](#)

Calling a RESTlet

Follow the OAuth 1.0 specification to generate a token. For more information and an example, see the help topic [Using OAuth in the Authorization Header](#).

Token-based Authentication and Web Services

Important: In past releases, TBA could not be used with the Administrator role. In 2018.1, this restriction has been removed. As of 2018.1, administrators can create access tokens for their own use. An administrator cannot create a token for another administrator.

Token-based Authentication (TBA) supports SuiteTalk (web services).

Token-based authentication decouples web services authentication from password expiration. Client applications can access web services using a token, significantly reducing the risk of compromising user credentials. For more information, see the help topic [Integration Management](#).

You enable token-based authentication by creating an integration record for an external application. On the record, you must check the Token-based Authentication box. For guidance on adapting an integration to include TBA credentials and to see an example that includes code snippets and SOAP headers, see the help topic [Token-Based Authentication Details](#).

With TBA, you use the `TokenPassport` complex type to send credentials. The `TokenPassport` references the `TokenPassportSignature` complex type, another important element in the token-based authentication process. See the help topic [TokenPassport Complex Type](#).

For more information about using token-based authentication with web services, see the following topics:

- [Requirements for Using Token-Based Authentication](#)
- [Regenerating a Consumer Key and Secret](#)
- [Web Services Governance for Token-Based Authentication](#)

TBA: Calling a Token Endpoint

Important: In past releases, TBA could not be used with the Administrator role. In 2018.1, this restriction has been removed. As of 2018.1, administrators can create access tokens for their own use. An administrator cannot create a token for another administrator. Tokens for Administrators can only be created by clicking the Manage Access Tokens link in the Settings portlet or by using the token endpoint.

In addition to creating a token manually through the NetSuite UI, developers and users can issue or revoke their own tokens programmatically using a token endpoint. Use the appropriate domain to call the token endpoint:

- RESTlet domain: `https://rest.netsuite.com/rest/`
- system domain: `system.netsuite.com/rest/`

Important: Users cannot programmatically issue or revoke tokens for other users using a token endpoint. For information about creating tokens for other users through the NetSuite UI, see [Viewing, Editing, Creating, and Revoking TBA Tokens](#).

Note: As of 2017.2, account-specific domains are supported for RESTlets, and you can access your RESTlet domain at the following URL, where 123456 is your account ID: `123456.restlets.api.netsuite.com`. The data center-specific domains supported before 2017.2 will continue to be supported. For more information, see the help topic [URLs for Account-Specific Domains](#).

Calling a token endpoint to issue a token

- Use the NetSuite `NLAuth` Authorization header. The token is created under the role specified in the `NLAuth` Authorization header. For more information, see the help topic [Using NLAuth in the Authorization Header](#).
- Parameters must be Url encoded. This is particularly important for parameters which include special characters like spaces, for example, token name.
- A token endpoint consumes two GET parameters. For an `issuetoken` request, the Consumer Key parameter is mandatory, and the Name (the name of the token) is optional.

For example:

```
https://rest.netsuite.com/rest/issuetoken?consumerKey=<CONSUMER_KEY>&name=<TOKEN_NAME>
```

Calling a token endpoint to revoke a token

- In a call to a token endpoint to revoke a token, use either:

- The NetSuite `NLAuth` Authorization header. See the help topic [Using NLAuth in the Authorization Header](#) in the topic [Authentication for RESTlets](#).
- The `OAuth` Authorization header. See the help topic [Using OAuth in the Authorization Header](#) in the topic [Authentication for RESTlets](#).
- A token endpoint consumes two GET parameters. For a `revoketoken` request, both the Consumer Key parameter and the Token parameter are mandatory.

Here is an example of a request: `https://rest.netsuite.com/rest/revoketoken?consumerKey=<CONSUMER_KEY>&token=<TOKEN>`

Two-Factor Authentication

Two-factor authentication (2FA) allows enforcement of an additional level of security for logging in to NetSuite. Using 2FA can protect your company from unauthorized access to data.

Two-factor authentication requires that users log in using:


1. Something they know - their NetSuite user credentials: their email address and password.
2. Something they have:
 - a secure token that generates a time-based verification code for each login, or
 - a mobile phone that can receive verification codes by:
 - Short Message Service (SMS) text message, or
 - voice call, or
 - an authenticator application

Each code is a unique series of numbers valid for a limited time, and only for a single login.

Types of Two-Factor Authentication Supported in NetSuite

The NetSuite application supports two different types of Two-Factor Authentication: 2FA by Phone or Authenticator App and 2FA Using RSA Tokens.

- Two-Factor Authentication by Phone or Authenticator Application, referred to as 2FA by Phone or Authenticator App. SMS text messages, voice calls, and third-party authenticator applications are all supported delivery methods for verification codes, however we recommend using an authenticator app. See [About 2FA by Phone or Authenticator App](#) for more information. See also [Supported Authenticator Apps](#).
- Two-Factor Authentication using RSA SecurID hardware and software tokens. See [About 2FA Using RSA Tokens](#) for more information.

 **Note:** As of September 1, 2017, we are no longer selling the Two-Factor Authentication feature that requires RSA tokens. Existing customers with this feature can continue to use RSA tokens and can purchase new RSA tokens as needed from their vendor of choice. Customers requiring two-factor authentication (2FA) for account access should consider using the 2FA solution built in to NetSuite. See [2FA by Phone or Authenticator App](#) for more information.

Both types of 2FA authentication supported for the NetSuite application can be used simultaneously in a single account. There are similarities between these two types of authentication, but the capabilities of each type differ.

Note: The Two-Factor Authentication feature is not compatible with SuiteTalk (web services) or SuiteAnalytics Connect. To use web services or SuiteAnalytics Connect, you must be logged in with a role that does not require 2FA. If you want to use RESTlets or web services with your 2FA-enabled role, use Token-based Authentication. See [Token-based Authentication](#) for more information.

About 2FA by Phone or Authenticator App

2FA by Phone options have been available in all NetSuite accounts for the past several releases. The 2FA by Authenticator App option was added in 2017.1.

Administrators: you do not have to enable a feature to use this type of 2FA in your NetSuite account. You do not have to purchase or upload tokens. Setup required of administrators is minimal. You can start using 2FA by Phone or Authenticator App in your NetSuite account whenever you want to get started. Account administrators, or other users with the **Two-Factor Authentication base** permission, must designate roles as 2FA authentication required, and users who are assigned to these roles must set up their phone numbers and authenticator applications in NetSuite.

 [View a related video.](#)

Reasons for choosing to use **2FA by Phone or Authenticator App** in your NetSuite account include:

- No special licensing is required. (No cost.)
- No special tokens are required. (No cost.)
- Access is supported for the NetSuite UI and NetSuite Mobile applications.
- Little maintenance is required of administrators. After being assigned to a 2FA authentication required role, users configure their own settings and manage their own devices in NetSuite.
- Works with all non-customer center roles, including contacts.
- The user's 2FA settings are shared across all NetSuite accounts and for all companies to which they have access.
- There are two authentication options available for users, and users can switch between these options when they log in:
 - The 2FA by Authenticator App option is recommended as the primary authentication method because it is always available. Even if a user cannot receive a text message (SMS) or a voice call, the authenticator app can generate a verification code. For a list of third-party authentication applications, see the help topic [Supported Authenticator Apps](#).
 - The 2FA by Phone option lets users specify their preferred delivery method for verification codes: Text message (SMS) or voice call. Users only need to set up their phone number in NetSuite and specify how they prefer to receive verification codes. If necessary, administrators can verify which delivery methods are available in your country. See [Supported Countries: SMS and Voice Call](#).

For more information about this type of two-factor authentication, see [2FA by Phone or Authenticator App](#).

About 2FA Using RSA Tokens

Note: As of September 1, 2017, we are no longer selling the Two-Factor Authentication feature that requires RSA tokens. Existing customers with this feature can continue to use RSA tokens and can purchase new RSA tokens as needed from their vendor of choice. Customers requiring two-factor authentication (2FA) for account access should consider using the 2FA solution built in to NetSuite. See [2FA by Phone or Authenticator App](#) for more information.

Important: See [RSA Required Roles in Release Preview and Sandbox Accounts](#) for more information.

To access NetSuite using RSA Tokens, you must have already purchased the **Two-Factor Authentication** feature from NetSuite. After purchase, a designated number of 2FA licenses was provisioned to your NetSuite account. Account administrators (or other users with the **Two-Factor Authentication** permission) must manage the RSA tokens in NetSuite.

- RSA SecurID hardware and software tokens are supported. You can acquire the token from the vendor of your choice.
- 2FA using RSA access is supported in NetSuite and NetSuite Mobile.
- 2FA using RSA works with all non-customer center roles, except contacts.

Note: Consider using 2FA by Phone or Authenticator App if 2FA is required for contacts.

- Up to three tokens can be assigned to a single user.
- With 2FA using RSA, 2FA settings are valid in one NetSuite company account and the primary sandbox account.

For more information about tasks that are performed by administrators, see [2FA Using RSA Tokens](#).

Managing Two-Factor Authentication

The following are required tasks for managing two-factor authentication (2FA) in a NetSuite account. These tasks can be completed by account administrators and by other users that have the Two-Factor Authentication permission.

- For roles that are to be restricted as 2FA roles, designate the type of 2FA authentication (RSA 2FA authentication required or 2FA authentication required). See [Designating Two-Factor Authentication Roles](#).
- When using 2FA by Phone or Authentication App, after administrators designate roles and assign them to users, the users:
 - Are sent a verification code by email during the initial login attempt to a 2FA role.
 - Must select their preferred authentication method.
 - If the user selects 2FA by Phone, they must register their phone number in NetSuite, which is tied to their email address.
 - If the user selects 2FA by Authenticator App, they must set up their chosen authenticator application. For a list of supported authenticator apps, see the help topic [Supported Authenticator Apps](#).
 - Select a backup 2FA method (optional).
 - Are provided ten backup codes, to be used when they are not able to receive a verification code through their authenticator app, text message (SMS), or a voice call.
- When using 2FA with RSA SecurID tokens:

Note: As of September 1, 2017, we are no longer selling the Two-Factor Authentication feature that requires RSA tokens. Existing customers with this feature can continue to use RSA tokens and can purchase new RSA tokens as needed from their vendor of choice. Customers requiring two-factor authentication (2FA) for account access should consider using the 2FA solution built in to NetSuite. See [2FA by Phone or Authenticator App](#) for more information.

Important: See also [RSA Required Roles in Release Preview and Sandbox Accounts](#) for more information.

- Token information must be uploaded into NetSuite from an XML file provided by the token vendor. This information includes the serial number and type for each token. See [Working with RSA Tokens in Multiple Account Types](#) and [Uploading RSA Token Information into NetSuite](#).
- After token information has been uploaded, tokens can be associated with users in NetSuite. All users with assigned 2FA roles designated as **RSA token authentication required** must have at least one hardware or software token to use for obtaining verification codes. See [Associating Users with RSA Tokens](#).

Important: When using RSA SecurID tokens, perform all tasks in your production account to ensure proper synchronization with the RSA server. Refresh a sandbox to make 2FA available there. Also, when logging in to accounts that share RSA tokens, a verification code can only be used a single time. For example, users must obtain a code to log into the production account and obtain a new code to log in to the sandbox account.

- A search capability makes it easy for administrators to find a particular token in NetSuite. See [Searching for an RSA Token](#).

Each time an employee logs in to NetSuite with a 2FA role, they must enter an email address and password followed by a verification code, obtained from their associated token, authenticator app, or from the response to their request for a text message (SMS) or voice call. See the help topic [Logging in Using Two-Factor Authentication \(2FA\)](#).

Note: The Two-Factor Authentication feature is not compatible with SuiteTalk (web services) or SuiteAnalytics Connect. To use web services or SuiteAnalytics Connect, you must be logged in with a role that does not require 2FA. If you want to use RESTlets or web services with your 2FA-enabled role, use Token-based Authentication. See [Token-based Authentication](#) for more information.

Designating Two-Factor Authentication Roles

Note: As of September 1, 2017, we are no longer selling the Two-Factor Authentication feature that requires RSA tokens. Existing customers with this feature can continue to use RSA tokens and can purchase new RSA tokens as needed from their vendor of choice. Customers requiring two-factor authentication (2FA) for account access should consider using the 2FA solution built in to NetSuite. See [2FA by Phone or Authenticator App](#) for more information.

Important: See also [RSA Required Roles in Release Preview and Sandbox Accounts](#) for more information.

An account administrator or another user with the Two-Factor Authentication permission can use the Two-Factor Authentication Roles page to indicate roles that require 2FA for login. Each 2FA role can be

configured to specify how often users with that role should be presented with the 2FA challenge. The default is per session, and the Duration of Trusted Device column includes values for hours (4, 6, 8, 12) and days (1–30). The value specified in the Duration of Trusted Device column works in conjunction with the devices users indicate as trusted devices. See [Users and Trusted Devices for Two-Factor Authentication](#) for more information.

Important: 2FA restrictions can be defined for most roles, including Employee Center, Partner Center, and Vendor Center roles, but not for Customer Center roles. Access for Contacts is not supported for 2FA using RSA tokens.

Note: When using RSA SecurID tokens, perform all tasks in your production account to ensure proper synchronization with the RSA server. Refresh a sandbox to make 2FA available there.

To designate two-factor authentication roles:

1. Go to Setup > Users/Roles > Two-Factor Authentication Roles.
2. For roles that require 2FA, select the type of authentication (RSA 2FA authentication required or 2FA authentication required) in the **Two-Factor Authentication Required** column.

Note: A single 2FA role can be designated as either RSA 2FA authentication required or 2FA authentication required, but not both. For example, if you wish to have a role that has both 2FA by Phone or Authenticator App and RSA 2FA capabilities, you must make a copy of the role and designate one as **2FA authentication required**, and designate the copy as **RSA 2FA authentication required**.

Two-Factor Authentication Roles		
Submit		
SHOW INACTIVES		TOTAL: 50
ROLE	TWO-FACTOR AUTHENTICATION REQUIRED	DURATION OF TRUSTED DEVICE
Accountant	Not required	Per session
Accountant (Reviewer)	Not required	Per session
Accountant TFA	2FA authentication required	Per session
Administrator	Not required	Per session
Advanced Partner Center	RSA 2FA authentication required	Per session
Bookkeeper	Not required	Per session
Bookkeeper TFA	2FA authentication required	4 Hours

3. In the **Duration of Trusted Device** column, accept the default (Per Session) or select the length of time before a device a user has marked as trusted will be subject to a two-factor authentication request.
4. Click **Submit**.

2FA roles designated as **RSA 2FA authentication required** are accessible only to users with associated tokens. Before tokens can be associated with users, token information must be uploaded into NetSuite. See [Uploading RSA Token Information into NetSuite](#).

RSA Required Roles in Release Preview and Sandbox Accounts

Customers using RSA SecurID for two-factor authentication (2FA) in their accounts should already be aware that they can only use their RSA tokens in accounts with the same account ID. Accounts such as

the 2018.1 Release Preview account, and sandbox accounts that are accessed from the same domain as the production account (system.netsuite.com) have a different account ID than the production account. Therefore, RSA tokens used for access to the production account will not work in a Release Preview account, or a sandbox account that is accessed from the same domain as the production account. Users with Administrator roles designated as RSA 2FA authentication required would be prevented from accessing Release Preview and sandbox accounts.



Important: To ensure access to 2018.1 Release Preview and sandbox accounts that are accessed from the system.netsuite.com domain, we have transitioned RSA required roles in those accounts from RSA 2FA authentication required to 2FA authentication required roles. This transition occurs when Release Preview accounts are created, and during the sandbox refresh process. These 2FA authentication roles are associated with the no-cost, built-in 2FA by Phone or Authenticator App option in NetSuite. The transition of RSA 2FA required roles ensures that users are able to access Release Preview and refreshed sandbox accounts, and maintains the extra security requirement of 2FA.

On the first attempt to access the 2FA role in your sandbox or Release Preview account, users will be prompted to set up the 2FA by Phone or Authentication App. For sandbox accounts, this transition of the RSA 2FA authentication required role to a 2FA authentication required occurs each time the sandbox is refreshed. For more information, see the following help topics:

- [Using 2FA by Phone or Authenticator App](#)
- [Designating Two-Factor Authentication Roles](#)
- [About Sandbox Accounts on the NetSuite Domain](#)
- [Working with RSA Tokens in Multiple Account Types](#)

Users and Trusted Devices for Two-Factor Authentication



Note: As of September 1, 2017, we are no longer selling the Two-Factor Authentication feature that requires RSA tokens. Existing customers with this feature can continue to use RSA tokens and can purchase new RSA tokens as needed from their vendor of choice. Customers requiring two-factor authentication (2FA) for account access should consider using the 2FA solution built in to NetSuite. See [2FA by Phone or Authenticator App](#) for more information.

Users with 2FA authentication required roles (2FA by Phone or Authenticator App, or 2FA using RSA) can specify devices as trusted when logging in to the 2FA role. Marking a device as trusted works in conjunction with the value specified by the Administrator in the Duration of Trusted Device column for a particular role.

For example, a role has been designated as 2FA required, and the value for Duration of Trusted Device has been set to 30 days. The next time a user with this role logs in, they can choose whether to select the Trust this device box.



Important: In cases where a user has access to more than one company, the results of marking a device as trusted varies by the type of 2FA authentication required by the role.

- If the user's role requires authentication with an RSA token, marking a device as trusted is valid only for that company. RSA tokens are valid for a specific company.

- If the user's role requires authentication using 2FA by Phone or Authenticator App, marking a device as trusted makes that device trusted across all companies to which the user has access.

ORACLE | NETSUITE

Two-Factor Authentication for Wolfe Electronics

Use your token to obtain a verification code. Enter your verification code below.

Verification code

☒ Trust this device for 30 days for access to this role.

Submit

[Lost your token?](#)

The user is in complete control of whether devices are considered trusted. In this example, the user could select the Trust this device box and then would not be presented with a 2FA challenge for 30 days when logging in to NetSuite from this device.

After a user has marked a device as trusted, the user can modify that choice on the Manage Trusted Devices page.

For example, this user marked a device as trusted. That is, the user previously chose not to be asked for a Two-Factor Authentication (2FA) verification code on this device.

The user can reverse that choice by selecting a **Restore 2FA required...** option. If the 2FA required is restored for a device, the user must have a 2FA token, authenticator app, phone, or a backup code to log in.

Activities Billing Customers Vendors Payroll and HR Financial Reports Documents Setup ...

Manage Trusted Devices

Submit **Cancel** [More](#)

THIS DEVICE IS MARKED AS TRUSTED. ON 6/23/2016, YOU CHOSE NOT TO BE CHALLENGED FOR A TWO-FACTOR AUTHENTICATION (2FA) ONE-TIME PASSWORD ON THIS DEVICE. YOU WILL ONLY NEED YOUR USERNAME AND PASSWORD TO SIGN IN ON A TRUSTED DEVICE.

CURRENT PASSWORD *

☐ RESTORE 2FA REQUIRED FOR ALL MY TRUSTED DEVICES

☒ RESTORE 2FA REQUIRED FOR THIS DEVICE

There is a link on the Settings portlet to access the Manage Trusted Devices page.

Settings

- Personalize Dashboard
- Set Preferences
- Campaign Subscription Center
- Change Email
- Change Password
- Update Security Questions
- Manage Trusted Devices**

2FA by Phone or Authenticator App

[View a related video.](#)

Note: The 2FA by Authenticator App option is available in your account, and is the recommended option for users with roles that are designated as **2FA authentication required**. It is not always possible for users to receive a text message (SMS) or voice call. Authenticator apps are always available for generating verification codes. For more information, see the help topic [Supported Authenticator Apps](#).

2FA by Phone or Authenticator App is available for all companies using NetSuite in all their NetSuite accounts as a method of improving security. 2FA by Phone or Authenticator App can help you to comply with IT security standards and regulations, using phones your users already have. Unlike 2FA using RSA Tokens, 2FA by Phone or Authenticator App is not tied to a single company in NetSuite. As long as the user's session remains valid, the user will not be asked again for a verification code when they switch between roles, even when switching between roles in different companies.

To use 2FA by Phone or Authenticator App, account administrators (or other users with the permission **Two-Factor Authentication base**) must designate specific roles as **2FA authentication required** roles. See [Designating Two-Factor Authentication Roles](#) for more information.

Each user assigned to a 2FA role designated as **2FA authentication required** must set up a phone number or authenticator application in NetSuite. When using 2FA by Phone, the user's phone number is linked to the email address they use to log in to the NetSuite UI.

After a role has been designated as **2FA authentication required**, a user assigned to that role receives an email the first time they attempt to login to the 2FA role. The email contains instructions and a verification code for initial login.

After completing the initial login to a 2FA role, a popup window opens allowing the user to select their preferred 2FA options for generating verification codes.

If you wish to read topics written for users, see the following:

- [Logging in Using Two-Factor Authentication \(2FA\)](#)
- [Logging in with 2FA by Phone or Authenticator App](#)

Important: **2FA by Phone in your Sandbox** — For sandbox accounts that are accessed from the `system.sandbox.netsuite.com` domain: When you refresh your sandbox, 2FA authentication required roles are copied from your production account to your sandbox. However, phone numbers are not copied and authenticator apps are not set up. Users will need to set up their phone numbers and authenticator apps in the sandbox account just as they did in the production account.

Resetting a User's 2FA Settings

Account administrators can reset 2FA settings for users who may be unable to log in and reset them on their own.

Note: 2FA by Phone or Authenticator App is not restricted to a single company. An account administrator can only reset a user's 2FA settings when the user has access to the same company accounts that the administrator manages. If the administrator can reset a user's password, the administrator should also have the ability to reset the user's 2FA settings.

To reset a user's 2FA settings:

1. As an Administrator, go to Setup > Users/Roles > Two-Factor Authentication > Two-Factor Reset Tool.
2. On the Reset 2FA Settings page, enter the **Email Address** of the user whose 2FA settings you want to reset.
3. Click **Reset**.




Important: If you receive an error message that the 2FA settings cannot be reset, it usually indicates that you do not have management over all the accounts that the user has access to under that email address. Click **Go Back** and click **Cancel**. Contact NetSuite Support for assistance with resetting the 2FA settings for the user.


4. A confirmation page states that the registered 2FA devices have been successfully reset.


Supported Countries: SMS and Voice Call

Although the phone number setup required for users is fairly straightforward, you or your users might have questions about the supported delivery methods available in your country for receiving verification codes.

The following table lists supported countries and the supported delivery methods for access to NetSuite.

Country	Supported Delivery Methods	Country Code
Afghanistan	SMS	+93
Åland Islands (Finland)	SMS Voice call	+358
<div>  Note: In the Set Up 2FA Phone Number popup window, Åland Islands is at the end of the list of countries. </div>		
Albania	SMS Voice call	+355
Algeria	SMS	+213
American Samoa	SMS	+1 Area Code: 684
Andorra	SMS Voice call	+376
Angola	SMS	+244
Anguilla	SMS	+1 Area Code: 264
Antigua and Barbuda	SMS	+1 Area Code: 268
Argentina	SMS Voice call	+54
Armenia	SMS Voice call	+374
Aruba	SMS	+297
Ascension	SMS	+247
Australia	SMS Voice call	+61
Austria	SMS Voice call	+43
Azerbaijan	SMS	+994


Country	Supported Delivery Methods	Country Code
	Voice call	
Bahamas	SMS	+1 Area Code: 242
Bahrain	SMS	+973
Bangladesh	SMS	+880
Barbados	SMS	+1 Area Code: 246
Belarus	SMS Voice call	+375
Belgium	SMS Voice call	+32
Belize	SMS	+501
Benin	SMS	+229
Bermuda	SMS	+1 Area Code: 441
Bhutan	SMS	+975
Bolivia	SMS	+591
Bosnia and Herzegovina	SMS Voice call	+387
Botswana	SMS	+267
Brazil	SMS Voice call	+55
British Virgin Islands	SMS Voice call	+1 Area Code: 284
Brunei	SMS	+673
Bulgaria	SMS Voice call	+359
Burkina Faso	SMS	+226
Burundi	SMS	+257
Cambodia	SMS	+855
Cameroon	SMS	+237
Canada	SMS Voice call	+1 Multiple Area Codes
Canary Islands	SMS	+3491
<div>  Note: In the Set Up 2FA Phone Number popup window, select Spain (+34) as the country. Type 91 then type your phone number. </div>		
Cape Verde	SMS	+238
Caribbean Netherlands (Netherlands Antilles)	SMS	+599

Country	Supported Delivery Methods	Country Code
Cayman Islands	SMS	+1 Area Code: 345
Central African Republic	SMS	+236
Chad	SMS	+235
Chile	SMS Voice call	+56
China	SMS Voice call	+86
Christmas Island	SMS Voice call	+61
Cocos (Keeling) Islands	SMS Voice call	+61
Colombia	SMS	+57
Comoros	SMS	+269
Congo, Democratic People's Republic	SMS	+243
Congo, Republic of	SMS	+242
Costa Rica	SMS	+506
Côte d'Ivoire (Ivory Coast)	SMS	+225
Croatia	SMS Voice call	+385
Cuba	SMS	+53
Cyprus	SMS Voice call	+357
Czech Republic	SMS Voice call	+420
Denmark	SMS Voice call	+45
Djibouti	SMS	+253
Dominica	SMS	+1 Area Code: 767
Dominican Republic	SMS	+1 Area Code: 809
East Timor	SMS	+670
<div>  Note: In the Set Up 2FA Phone Number popup window, this country is Timor-Leste in the list. </div>		
Ecuador	SMS	+593
Egypt	SMS Voice call	+20
El Salvador	SMS	+503

Country	Supported Delivery Methods	Country Code
Equatorial Guinea	SMS	+240
Eritrea	SMS	+291
Estonia	SMS Voice call	+372
Ethiopia	SMS	+251
Falkland Islands	SMS	+500
Faroe Islands	SMS Voice call	+298
Fiji	SMS Voice call	+679
Finland (including the Åland Islands)	SMS Voice call	+358
France	SMS Voice call	+33
French Guiana	SMS	+594
French Polynesia	SMS	+689
Gabon	SMS	+241
Gambia	SMS	+220
Georgia	SMS Voice call	+995
Germany	SMS Voice call	+49
Ghana	SMS	+233
Gibraltar	SMS	+350
Greece	SMS Voice call	+30
Greenland	SMS	+299
Grenada	SMS	+1 Area Code: 437
Guadeloupe	SMS	+590
Guam	SMS	+1 Area Code: 671
Guatemala	SMS	+502
Guernsey (United Kingdom)	SMS Voice call	+44
Guinea	SMS	+224
Guinea-Bissau	SMS	+245
Guyana	SMS	+592
Haiti	SMS Voice call	+509

Country	Supported Delivery Methods	Country Code
Honduras	SMS	+504
Hong Kong	SMS Voice call	+852
Hungary	SMS Voice call	+36
Iceland	SMS Voice call	+354
India	SMS Voice call	+91
Indonesia	SMS Voice call	+62
Iran	SMS	+98
Iraq	SMS	+964
Ireland	SMS Voice call	+353
Isle of Man (United Kingdom)	SMS Voice call	+44
Israel	SMS Voice call	+972
Italy	SMS Voice call	+39
Jamaica	SMS	+1 Area Code: 876
Japan	SMS Voice call	+81
Jersey (United Kingdom)	SMS Voice call	+44
Jordan	SMS	+962
Kazakhstan	SMS Voice call	+7
Kenya	SMS	+254
Kosovo	SMS	+883
Kuwait	SMS Voice call	+965
Kyrgyzstan	SMS Voice call	+996
Laos	SMS	+856
Latvia	SMS Voice call	+371
Lebanon	SMS	+961
Lesotho	SMS	+266
Liberia	SMS	+231

Country	Supported Delivery Methods	Country Code
Libya	SMS	+218
Liechtenstein	SMS Voice call	+423
Lithuania	SMS Voice call	+370
Luxembourg	SMS Voice call	+352
Macau	SMS Voice call	+853
Macedonia	SMS Voice call	+389
Madagascar	SMS	+261
Malawi	SMS	+265
Malaysia	SMS Voice call	+60
Maldives	SMS	+960
Mali	SMS	+223
Malta	SMS Voice call	+356
Marshall Islands	SMS	+692
Martinique	SMS	+596
Mauritania	SMS	+222
Mauritius	SMS	+230
Mayotte	SMS	+262
Mexico	SMS Voice call	+52
Micronesia	SMS	+691
Moldova	SMS Voice call	+373
Monaco	SMS Voice call	+377
Mongolia	SMS	+976
Montenegro	SMS Voice call	+382
Montserrat	SMS	+1 Area Code: 664
Morocco	SMS	+212
Mozambique	SMS	+258
Myanmar	SMS	+95
Namibia	SMS	+264

Country	Supported Delivery Methods	Country Code
Nepal	SMS	+977
Netherlands	SMS Voice call	+31
Netherlands Antilles (Caribbean Netherlands)	SMS	+599
 Note: In the Set Up 2FA Phone Number popup window, this country is listed as Caribbean Netherlands.		
New Caledonia	SMS	+687
New Zealand	SMS Voice call	+64
Nicaragua	SMS	+505
Niger	SMS	+227
Nigeria	SMS	+234
North Korea	SMS	+850
Northern Mariana Islands	SMS	+1 Area Code: 670
Norway	SMS Voice call	+47
Oman	SMS	+968
Pakistan	SMS Voice call	+92
Palau	SMS	+680
Palestinian Territory (Palestine)	SMS	+970
Panama	SMS Voice call	+507
Papua New Guinea	SMS	+675
Paraguay	SMS Voice call	+595
Peru	SMS Voice call	+51
Philippines	SMS Voice call	+63
Poland	SMS Voice call	+48
Portugal	SMS Voice call	+351
Puerto Rico	SMS Voice call	+1 Area Codes: 787, 939
Qatar	SMS Voice call	+974
Réunion Island	SMS	+262

Country	Supported Delivery Methods	Country Code
Romania	SMS Voice call	+40
Russia (Russian Federation)	SMS Voice call	+7
Rwanda	SMS	+250
Saint Barthélemy	SMS	+590
Saint Kitts and Nevis	SMS	+1 Area Code: 869
Saint Lucia	SMS	+1 Area Code: 758
Saint Martin (French side)	SMS	+590
Saint Pierre and Miquelon	SMS	+508
Saint Vincent and the Grenadines	SMS	+1 Area Code: 784
Samoa	SMS	+685
San Marino	SMS Voice call	+378
São Tomé and Príncipe	SMS	+239
Saudi Arabia	SMS Voice call	+966
Senegal	SMS	+221
Serbia	SMS Voice call	+381
Seychelles	SMS	+248
Sierra Leone	SMS	+232
Singapore	SMS Voice call	+65
Slovakia (Slovak Republic)	SMS Voice call	+421
Slovenia	SMS Voice call	+386
Solomon Islands	SMS	+677
Somalia	SMS	+252
South Africa	SMS Voice call	+27
South Korea	SMS Voice call	+82
South Sudan	SMS	+211
Spain	SMS Voice call	+34

Country	Supported Delivery Methods	Country Code
Sri Lanka	SMS	+94
Sudan	SMS	+249
Suriname	SMS	+597
Svalbard and Jan Mayen (Norway)	SMS Voice call	+47
Swaziland	SMS	+268
Sweden	SMS Voice call	+46
Switzerland	SMS Voice call	+41
Syria (Syrian Arab Republic)	SMS	+963
Taiwan	SMS Voice call	+886
Tajikistan	SMS Voice call	+992
Tanzania	SMS	+255
Thailand	SMS Voice call	+66
Timor-Leste (East Timor)	SMS	+670
Togo	SMS	+228
Tonga	SMS	+676
Trinidad and Tobago	SMS	+1 Area Code: 868
Tunisia	SMS	+216
Turkey	SMS Voice call	+90
Turkish Republic of Northern Cyprus	SMS	+90
Turkmenistan	SMS Voice call	+993
Turks and Caicos Islands	SMS	+1 Area Code: 649
Tuvalu	SMS	+688
U.S. Virgin Islands	SMS Voice call	+1 Area Code: 340
Uganda	SMS	+256
Ukraine	SMS Voice call	+380
United Arab Emirates	SMS Voice call	+971

Country	Supported Delivery Methods	Country Code
United Kingdom	SMS Voice call	+44
United States	SMS Voice call	+1 Multiple Area Codes
Uruguay	SMS Voice call	+589
Uzbekistan	SMS Voice call	+998
Vanuatu	SMS	+678
Vatican City	SMS Voice call	+379
Venezuela	SMS Voice call	+58
Vietnam	SMS Voice call	+84
Virgin Islands, British	SMS Voice call	+1 Area Code: 284
Virgin Islands, U.S.	SMS Voice call	+1 Area Code: 340
Western Sahara	SMS	+212
Yemen	SMS	+967
Zambia	SMS	+260
Zimbabwe	SMS	+263

2FA Using RSA Tokens

Note: As of September 1, 2017, we are no longer selling the Two-Factor Authentication feature that requires RSA tokens. Existing customers with this feature can continue to use RSA tokens and can purchase new RSA tokens as needed from their vendor of choice. Customers requiring two-factor authentication (2FA) for account access should consider using the 2FA solution built in to NetSuite. See [2FA by Phone or Authenticator App](#) for more information.

Important: See also [RSA Required Roles in Release Preview and Sandbox Accounts](#) for more information.

Before your users can access NetSuite using RSA Tokens, you can purchase the **Two-Factor Authentication** feature from NetSuite. After this purchase, a designated number of 2FA licenses are provisioned to your NetSuite account. manage the RSA tokens in NetSuite.

Account administrators (or other users with the **Two-Factor Authentication** permission) must perform the following tasks:

- Purchase RSA SecurID hardware and software tokens. You can acquire them from the vendor of your choice. See [Purchasing RSA Tokens](#).

- Determine how you want to distribute tokens in your NetSuite accounts. That is, you might need to distribute different tokens in your production account and your primary sandbox account. See [Working with RSA Tokens in Multiple Account Types](#) for strategies for managing tokens.
- Upload the tokens (also called the seed file) to NetSuite. See [Uploading RSA Token Information into NetSuite](#).
- Designate roles as **RSA token authentication required** roles. See [Designating Two-Factor Authentication Roles](#).
- Determine the length of time you will allow devices to be trusted. See [Users and Trusted Devices for Two-Factor Authentication](#).
- Assign RSA token authentication required roles to users. See the help topic [NetSuite Users Overview](#) for more information.
- Users must have at least one hardware or software token to use for obtaining verification codes. Up to three tokens can be assigned to a single user. See [Associating Users with RSA Tokens](#).
- With 2FA using RSA, 2FA settings are valid in one NetSuite company account and the primary sandbox account.

Purchasing RSA Tokens

Note: As of September 1, 2017, we are no longer selling the Two-Factor Authentication feature that requires RSA tokens. Existing customers with this feature can continue to use RSA tokens and can purchase new RSA tokens as needed from their vendor of choice. Customers requiring two-factor authentication (2FA) for account access should consider using the 2FA solution built in to NetSuite. See [2FA by Phone or Authenticator App](#) for more information.

NetSuite supports RSA SecurID hardware and software tokens. You can acquire the hardware or software tokens that you need from the vendor of your choice. RSA requires token vendors to request an RSA license number.

- The NetSuite License Number for RSA is: **99764947**. You can provide this number if your token vendor requests it.
- Software tokens must be used in conjunction with the RSA Token app installed on users' mobile devices.

In addition to designating 2FA roles as **RSA Token authentication required**, account administrators must complete several tasks before 2FA using RSA Tokens can be used in your account.

If you wish to read topics written for users, see the following:

- [Logging in Using Two-Factor Authentication \(2FA\)](#)
- [Logging in with an RSA Token](#)

Working with RSA Tokens in Multiple Account Types

Note: As of September 1, 2017, we are no longer selling the Two-Factor Authentication feature that requires RSA tokens. Existing customers with this feature can continue to use RSA tokens and can purchase new RSA tokens as needed from their vendor of choice. Customers requiring two-factor authentication (2FA) for account access should consider using the 2FA solution built in to NetSuite. See [2FA by Phone or Authenticator App](#) for more information.

Important: See also [RSA Required Roles in Release Preview and Sandbox Accounts](#) for more information.

As a NetSuite account administrator, you are probably responsible for purchasing RSA tokens and uploading them into NetSuite. The usual practice is to purchase bundle of tokens from a token vendor – as many tokens as need to ensure you have enough for all users who will need to log in by 2FA using RSA. The bundle of tokens is provided from the vendor as a single .xml file containing the token serial numbers and token type information. This .xml file is sometimes called the seed file.

If you are the person responsible for ordering and uploading the tokens, you need to understand the implications of working with RSA tokens in different types of accounts. The following example illustrates a common situation, and potential solutions.

Example Use Case: Managing RSA Tokens in Multiple Account Types

Important: See [RSA Required Roles in Release Preview and Sandbox Accounts](#) for more information.

In the following scenario, you are the NetSuite account administrator for all the types of NetSuite accounts for your company: a production account, sandbox accounts, and a Release Preview account. If you are the administrator for accounts running in North American data centers that are accessed from separate domains, you might be the administrator for your company's production account and several sandboxes—a primary sandbox account and secondary sandbox account. You must ensure that token assignments and reassignments in one account do not affect the tokens in other accounts.

The recommended method of ensuring that token assignments in one account do not affect other accounts is to upload the seed file to your production account, and then refresh your primary sandbox account. When the refresh is complete, the primary sandbox includes the tokens and the token assignments from the production account.

Important: When using RSA SecurID tokens, perform all tasks in your production account to ensure proper synchronization with the RSA server. Refresh a sandbox to make 2FA available there. Also, when logging in to accounts that share RSA tokens, a verification code can only be used a single time. For example, users must obtain a verification code to log into the production account and obtain a new code to log in to the sandbox account.

In some situations, you might want to use some of the tokens in your secondary sandbox account. In this case, you could upload the seed file manually (as there is no way to push the seed file from production to secondary sandboxes) and assign some of the tokens to users in the secondary sandbox.

Warning: Exercise caution and assign only tokens that were not assigned in your production and primary sandbox accounts. If you assign tokens in a secondary sandbox that are already in use in those other accounts, you risk corrupting the token setup in your production account and primary sandbox.

There are several ways to avoid potential problems.

- You could choose to treat the tokens in the seed file as if there were an imaginary division in the file. Assign the first group of tokens only in production, and refresh your primary sandbox to use those tokens in the primary sandbox. Assign the second group of tokens only in the secondary sandbox. You will need to define your own process for managing this imaginary division of the tokens from the single seed file. All of the tokens in the seed file will be visible in all of the accounts in which you upload the file.
- Another, and certainly safer, alternative is to order two different seed files from the token vendor. Upload one seed file to the production account, and refresh your primary sandbox to use the tokens in the primary sandbox. Upload the other seed file to the secondary sandbox account, and assign those tokens in the secondary sandbox.

Best Practices for Managing RSA Tokens in Multiple Account Types

There are two methods for accessing sandbox accounts.

- Some sandboxes are accessed from the sandbox domain, `system.sandbox.netsuite.com`.
- Some sandboxes are accessed from the NetSuite domain, `system.netsuite.com`. Users access these sandboxes by switching from a production role to a sandbox role.



Important: See [RSA Required Roles in Release Preview and Sandbox Accounts](#) for more information.

Some customers have more than one sandbox account.

- For accounts accessed from the sandbox domain, customers can have a primary sandbox and a secondary sandbox (or secondary sandboxes).
- For accounts accessed from the NetSuite domain, there is no concept of a primary sandbox, that is, a sandbox with the same account ID as the production account. All sandboxes on the NetSuite domain have a unique account ID, and these sandboxes behave like secondary sandboxes.

Follow these guidelines for managing tokens:

- Do not assign the same tokens (that is, use the same seed file) in more than one account, such as in your production account and a secondary sandbox account, or in two sandbox accounts.
- You can use the same token setup in a primary sandbox, but understand that if you want to make any changes, do must make the changes in your production account and then propagate the changes to your primary sandbox by refreshing the sandbox.

Uploading RSA Token Information into NetSuite



Note: As of September 1, 2017, we are no longer selling the Two-Factor Authentication feature that requires RSA tokens. Existing customers with this feature can continue to use RSA tokens and can purchase new RSA tokens as needed from their vendor of choice. Customers requiring two-factor authentication (2FA) for account access should consider using the 2FA solution built in to NetSuite. See [2FA by Phone or Authenticator App](#) for more information.



Important: See also [RSA Required Roles in Release Preview and Sandbox Accounts](#) for more information.

An account administrator or another user with the **Two-Factor Authentication** permission can use the Two-Factor Authentication Tokens page to upload token serial number and token type information into NetSuite from an XML file provided by the token vendor. This upload must occur before users can be associated with tokens.

Note: When using RSA SecurID tokens, perform all tasks in your production account to ensure proper synchronization with the RSA server. Refresh your sandbox to make 2FA available in your sandbox.

To upload RSA token information:

1. Go to Setup > Users/Roles > Two-Factor Authentication Tokens.
2. Click **Upload**.
3. On the Two-Factor Token Authentication Token Upload page, browse to the XML file containing token data.
4. If you received a password with the XML file, enter that password.
5. Click **Submit**.

Token details are displayed on the list page.

Two-Factor Authentication Tokens		
Upload		
<div>FILTERS</div> <div> <div>User</div> <div>Token Serial Number</div> </div>		
<input type="checkbox"/> Show Inactives		
EDIT	TOKEN SERIAL NUMBER	TOKEN TYPE
Edit	000134821991	RSA SecurID 700 Authenticator
Edit	000134821992	RSA SecurID 700 Authenticator
Edit	000134821993	RSA SecurID 700 Authenticator
Edit	000134821994	RSA SecurID 700 Authenticator
Edit	000134821995	RSA SecurID 700 Authenticator
Edit	000147244633	RSA SecurID Software Authenticator
Edit	000147244634	RSA SecurID Software Authenticator
Edit	000147244635	RSA SecurID Software Authenticator
Edit	000147244636	RSA SecurID Software Authenticator
Edit	000147244637	RSA SecurID Software Authenticator

After token data has been uploaded into NetSuite, users can be associated with tokens. See [Associating Users with RSA Tokens](#).

Note: Access for Contacts is not supported for 2FA using RSA Tokens.


Associating Users with RSA Tokens

Note: As of September 1, 2017, we are no longer selling the Two-Factor Authentication feature that requires RSA tokens. Existing customers with this feature can continue to use RSA tokens and can purchase new RSA tokens as needed from their vendor of choice. Customers requiring two-factor authentication (2FA) for account access should consider using the 2FA solution built in to NetSuite. See [2FA by Phone or Authenticator App](#) for more information.

After token information has been uploaded, tokens can be associated with users in NetSuite. An account administrator or another user with the Two-Factor Authentication (2FA) permission can associate tokens with users in NetSuite. For details, see the following:

- [Associating Users with RSA Hardware Tokens](#)
- [Associating Users with RSA Software Tokens](#)
- [Associating Multiple RSA Tokens per User](#)
- [Steps for Associating an RSA Token with a User](#)


Notes for Customers with Sandbox Accounts: A user can use the same token for production and primary sandbox accounts. The user should be associated with the token in the production account first. You need to be careful to keep the association of users with tokens in production accounts and primary sandbox accounts in sync. Perform all user token setup in the production account first.

 **Note:** When using RSA SecurID tokens, perform all tasks in your production account to ensure proper synchronization with the RSA server. Refresh a sandbox to make 2FA available there.

Associating Users with RSA Hardware Tokens

An administrator can associate a hardware token with a user in NetSuite at the time when the physical token (the fob) is provided to the user, by selecting the user on the Two-Factor Authentication Token page. See [Steps for Associating an RSA Token with a User](#).

Or, a hardware token can be provided to a user without any entry in NetSuite. In this case, the user can activate that token on their first 2FA login and the token is then associated with that user. See the help topic [Activating Your Hardware Token](#).

 **Note:** Access for Contacts is not supported for 2FA using RSA Tokens.


Associating Users with RSA Software Tokens

A user cannot access a software token on a mobile device unless the RSA SecurID app is first installed on the device. Currently, the following mobile device types are supported for 2FA using RSA Token access:

- **iPhone or iPad** - See the help topic [Setting Up an RSA Token on iPhone or iPad](#).
- **Android** - See the help topic [Setting Up an RSA Token on Android](#).
- **BlackBerry** - See the help topic [Setting Up an RSA Token on BlackBerry](#).

To provide a software token to a user, on the Two-Factor Authentication Token page in NetSuite, an administrator selects the user, indicates the type of mobile device to be used for access, and enables an option to distribute the token. See [Steps for Associating an RSA Token with a User](#).

When the option to distribute a software token is enabled, an email that contains a software token file is sent to the user. This file can be opened in the RSA SecurID app to view a verification code for the user's first 2FA login. For each subsequent login, the user can open the RSA SecurID app to obtain a verification code.

 **Important:** Each email that delivers a software token is device-specific. Ensure that users do not assume that an email will work across different types of mobile devices. Also, when logging in to accounts that share RSA tokens, a verification code can only be used a single time. For example, users must obtain a code to log into the production account and obtain a new code to log in to the sandbox account.

Associating Multiple RSA Tokens per User

All users with assigned 2FA roles must have at least one hardware or software token to use for obtaining verification codes. In some cases, a single user may require more than one token. For

example, a user might want a hardware token to use generally and a software token to use for mobile device access.

A user can have up to three associated tokens. An error occurs if an administrator attempts to associate more than three tokens with a single user, or if a user attempts to use more than three tokens.

Steps for Associating an RSA Token with a User

1. Go to Setup > Users/Roles > Two-Factor Authentication Tokens.
2. Click the **Edit** link for the token you want to associate with a user.
3. On the Two-Factor Authentication page, select the user you want to associate with the token.

Two-Factor Authentication Token

[Submit](#) [Cancel](#)

Token Serial Number
000134821993

Token Type
RSA SecurID 700 Authenticator

User

☐ Inactive

Last Login
11/25/2013 7-12

Expires
12/31/2013 0-00

4. (Software tokens only) Check the **Distribute Token File** box and choose the type of mobile device to be used for 2FA access.

Two-Factor Authentication Token

Token Serial Number
000123456001

Token Type
RSA SecurID 700 Authenticator

User
[User Name] ▼

☐ Inactive

Last Login

Expires
1/30/2017 12:06 am

Software Token Distribution

☒ Distribute Token File

Software Token Device Type
[Android] ▼
Android
BlackBerry
iPhone

If you check the **Distribute Token File** box without selecting a user, no token file is distributed.

Note: The **Software Token Device Type** field is only used to select the proper token file format when **Distribute Token File** is checked. This value is not stored in the database. If you edit a particular software token (and, for example, change this value from Android to BlackBerry) the list will still display the original selection.

5. Click **Submit**.

Note the following about token user associations:

- A lost hardware token can be marked inactive.
- A token returned by a terminated employee can be reused by another employee. The user needs to be changed on the token record.

Note: Access for Contacts is not supported for 2FA using RSA Tokens.

Searching for an RSA Token

There are several methods available to search for two-factor authentication RSA tokens.

How to Search for RSA Tokens

You can search for a particular token on the Two-Factor Tokens Search page by serial number, associated user, expiration dates, and token types. This capability can be especially useful if you have a large number of tokens uploaded into NetSuite.

To search for a token:

1. Go to Setup > Users/Roles > Two-Factor Authentication Tokens.
2. Click **Search** in the upper right corner of the page.
3. On the Two-Factor Tokens page, enter your desired search parameters in the available fields.
 - For more information on entering search parameters, see the help topic [Defining a Simple Search](#).
 - If you need help in defining filters for a simple search, see the help topic [Tips for Defining Simple Search Filters](#).
 - If you need help with date filters, see the help topic [Defining Search Date Filters](#).
 - If you need more search criteria, check the **Use Advanced Search** box.
If you need help, see the help topic [Defining an Advanced Search](#).
4. Click **Submit**.

You can also search for a particular token on the Two-Factor Authentication Tokens list page by serial number or associated user.

To search RSA tokens by user:

1. Go to Setup > Users/Roles > Two-Factor Authentication Tokens.
2. In the **User** list, select the user associated with the token you want to find.
The page refreshes to display only the token(s) associated with the selected user.
To redisplay all tokens, reset the **User** list to be blank.

To search RSA tokens by serial number:

1. Go to Setup > Users/Roles > Two-Factor Authentication Tokens.
2. In the **Token Serial Number** box, enter all or part of the serial number for the token you want to find, and press return.
If you have entered the entire serial number, the page refreshes to display only that token.
If you have entered part of a serial number, the page refreshes to display only the tokens with serial numbers that contain the numbers you entered.
To redisplay all tokens, clear the **Token Serial Number** box and press **Return**.

Device ID Authentication

Device ID Authentication allows account administrators to restrict login to only approved devices. Devices can be registered in NetSuite using a unique identifier. After reviewing registered devices, the account administrator can approve or reject individual devices. Only devices approved by the administrator can log in.

The Device ID feature is enabled by default. No special setup or configuration in NetSuite by account administrators is required to use the device record.

See the following for information on using this feature:

- [Device ID and the SCIS SuiteApp](#)
- [Managing Devices on the List of devices Page](#)
- [The Device Record](#)
- [Creating Device Records Manually](#)
- [Viewing System Notes](#)
- [Deleting a Device Record](#)

Device ID and the SCIS SuiteApp

Currently, the Device ID feature is intended for use with Suite Commerce InStore (SCIS) SuiteApp and the point-of-sale (POS) devices running the SCIS POS application.

For more information on SCIS, see the help topic [SuiteCommerce InStore Administrator's Overview](#). See also, [Installing SCIS Mobile Apps](#).

Device records are automatically created in NetSuite as users log in for the first time using POS devices with the SCIS POS application installed. Users are then notified that they must wait for the device to be approved before they can use SCIS on that device. NetSuite account administrators maintain a list of approved POS devices in NetSuite. Only the devices that have been reviewed and approved have access to SCIS.

When the SuiteCommerce InStore SuiteApp is installed in a NetSuite account, it automatically creates a role restricted by device ID. This is the only role allowed to log in to SCIS on a device configured with the SCIS POS application.

Account administrators can create additional device ID restricted roles if desired, using the SCIS-created roles as a template. For more information on SCIS roles, see the help topic [SCIS Roles and Permissions](#).

The first time a user attempts to log in to SCIS on a device running the SCIS POS application, a unique device identifier is sent to NetSuite. The name of the device is also sent. With this information, a Device record is created in NetSuite in Pending status. Users cannot log in to SCIS with the device until the NetSuite account administrator reviews the device record and changes the device status to Trusted. This requirement ensures that only devices approved by the NetSuite account administrator can be used to log in. The account administrator can also change the device status to block a device, or put a device record on hold.

Managing Devices on the List of devices Page

With SCIS and the SCIS POS application, device records are automatically created in NetSuite as users log in from the POS devices for the first time. From the List of devices page, account administrators

maintain and manage the list of the POS devices that can potentially access the SCIS website in your NetSuite account.

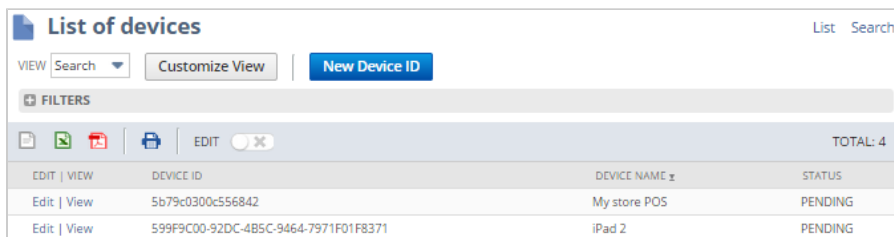
The List of devices page

To view the List of devices page, go to Setup > Integration > Device ID.

The following screenshot is an example of two device records that were created automatically when the users logged in from POS devices for the first time. (The SCIS POS application passed in the device ID and the device name when the user attempted to log in with the device.)

Both devices are in Pending status, and the device ID is displayed in full.

Important: To allow the account administrator the opportunity to verify the device, the device ID is displayed in full. As soon as the account administrator changes the status, the device ID is masked and cannot be retrieved from the system.

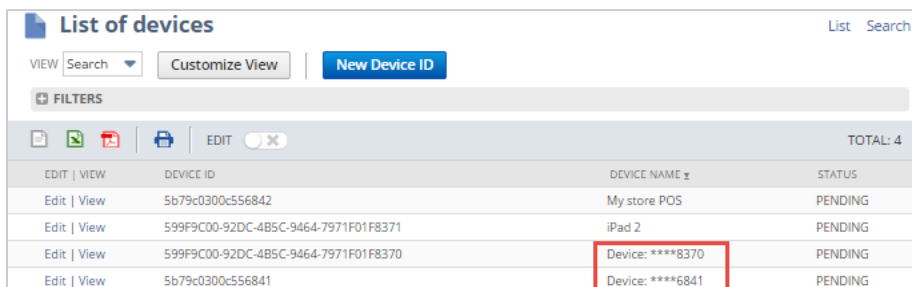


List of devices			
VIEW Search		Customize View	New Device ID
FILTERS			
EDIT VIEW			TOTAL: 4
EDIT VIEW	DEVICE ID	DEVICE NAME	STATUS
Edit View	5b79c0300c556842	My store POS	PENDING
Edit View	599F9C00-92DC-4B5C-9464-7971F01F8371	iPad 2	PENDING

Note: After the device record has been created, NetSuite recommends you do not change the device name on the device. If the name is changed on the device, administrators would need to make the corresponding update to the device record in NetSuite. The device record in NetSuite is never updated by the POS application on the device after the initial login creates the record.

In the following screenshot, the account administrator has created two additional device records manually. (See [Creating Device Records Manually](#) for more information.)

In this example, the administrator did not provide the optional Device Name when creating the records, and changed the device status to Pending before saving each record. NetSuite automatically created a Device Name using the Device ID, masking everything except the last four digits.



List of devices			
VIEW Search		Customize View	New Device ID
FILTERS			
EDIT VIEW			TOTAL: 4
EDIT VIEW	DEVICE ID	DEVICE NAME	STATUS
Edit View	5b79c0300c556842	My store POS	PENDING
Edit View	599F9C00-92DC-4B5C-9464-7971F01F8371	iPad 2	PENDING
Edit View	599F9C00-92DC-4B5C-9464-7971F01F8370	Device: ****8370	PENDING
Edit View	5b79c0300c556841	Device: ****6841	PENDING

The following screenshot is an example of the account administrator reviewing the List of devices, and changing the status of each device. The account administrator is sure the Device IDs for the automatically created records are correct, and changes the status to Trusted.

NEW	EDIT VIEW	DEVICE ID	DEVICE NAME	STATUS
	Edit View	5b79c0300c556842	My store POS	TRUSTED
	Edit View	599F9C00-92DC-4B5C-9464-7971F01F8371	iPad 2	TRUSTED
	Edit View	599F9C00-92DC-4B5C-9464-7971F01F8370	Device: ****8370	PENDING
	Edit View	5b79c0300c556841	Device: ****6841	ON_HOLD

For the manually created records, the account administrator wants more time to verify the Device ID numbers are correct, and changes the status for these records to On Hold. The following screenshot shows the list of devices after the statuses were changed, and the page was refreshed. All of the Device IDs have been masked, except for the last four digits.

Important: The only time the Device IDs are shown in full is when a device remains in the status in which it was initially created. This allows the account administrator to verify the Device ID. Treat Device IDs as securely as you would treat a password.

As soon as the device status is changed, the Device ID is masked, and cannot be retrieved from the system.

NEW	EDIT VIEW	DEVICE ID	DEVICE NAME	STATUS
	Edit View	****6842	My store POS	TRUSTED
	Edit View	****8371	iPad 2	TRUSTED
	Edit View	****8370	Device: ****8370	ON_HOLD
	Edit View	****6841	Device: ****6841	ON_HOLD

The Device Record

Account administrators can review the list of device records in their NetSuite account. Go to Setup > Integration > Device ID to access the List of devices page.

- To open a record, click Edit or View in the appropriate row on the List of devices page.
- To create a device record manually, click New Device ID. See [Creating Device Records Manually](#).

Note: An alternative method of creating a new device manually is available on the Device page. Select **New** from the **Actions** list to open a blank Device record.

- To view the System Notes for a device, see [Viewing System Notes](#)
- To delete a device record, see [Deleting a Device Record](#).

Creating Device Records Manually

It is possible for account administrators to create device records manually, but this approach is not recommended due to the potential for data entry errors. Before creating the record, ensure you have access to the correct device ID and device name.

To create a device record:

1. Go to Setup > Integration > Device ID.
2. Click **New Device ID**. The **Device** tab is displayed by default.
3. Enter the **Device Name**. If you do not enter a device name, a name will be generated automatically.
4. Enter the **Device ID**. The device ID should be a unique identifier for a specific device.
5. Manually created records default to the **Device Status** of Trusted, meaning the device is allowed to log in to NetSuite.

Change the status, if desired. Devices in any status other than Trusted will not be allowed to log in to NetSuite.

- **Pending:** Awaiting review and approval from an account administrator. For device records created manually, NetSuite recommends changing the status to Pending before saving the record. This allows additional time to verify the information (especially the Device ID) has been entered correctly.
 - **On Hold:** Has been reviewed by the account administrator, but is not yet approved. This status could be used for devices for a store that is not yet open, for example.
 - **Denied:** Reviewed by the account administrator and denied access to NetSuite.
6. Click **Submit**.

Viewing System Notes

Account administrators can view the change history of a device record on the System Notes tab.

To view the system notes for a device:

1. Go to Setup > Integration > Device ID.
2. On the List of devices page, click **Edit** or **View** for a particular device.
3. On the Device page, click the **System Notes** tab.

DATE	SET BY	CONTEXT	TYPE	FIELD	OLD VALUE	NEW VALUE
9/3/2015 9:41 am	-System-		Change	Device id	***	***
9/3/2015 9:41 am	-System-		Change	Device id hash	***	***
9/3/2015 9:41 am	-System-		Set	Device name		HouseWaresPOS_Store45
9/3/2015 9:41 am	-System-		Set	Status		PENDING
9/3/2015 9:44 am	-System-		Change	Device id	***	***
9/3/2015 9:44 am	-System-		Change	Status	PENDING	TRUSTED

Deleting a Device Record

Account administrators can delete a device record. For example, you may want to delete a device that was created manually if information such as the device ID was not entered correctly.

To delete a device record:

1. Go to Setup > Integration > Device ID.
2. On the List of devices page, click **Edit** or **View** for a particular device.
3. Select **Delete** from the **Actions** list.

4. If you are sure you want to delete this device, click **OK**.
A confirmation notice displays on the List of devices page.



Note: You can search for a device's login history using the Login Audit Trail search capabilities, even after the device record has been deleted. See the help topic [Login Audit Trail Overview](#). Only a device that has been used for login leaves a login audit trail. Searching for a device that was never used for login will have no results.

Outbound Single Sign-on (SuiteSignOn)

SuiteSignOn provides seamless integration of NetSuite with other applications through outbound single sign-on. With this feature, NetSuite users can access external applications directly from the NetSuite user interface without additional authentication.

Anyone using the SuiteSignOn feature should see the following topics:

- [SuiteSignOn Overview](#)
- [Understanding SuiteSignOn](#) for a conceptual overview of SuiteSignOn.
- [SuiteSignOn Required Features](#), as a SuiteSignOn solution cannot be implemented until certain features are enabled.

Application providers who want to sell their applications and services to customers who also use NetSuite must see these topics as well:

- [Setting Up SuiteSignOn Integration](#)
- [Creating a SuiteSignOn Bundle](#)
- [SuiteSignOn Code Samples](#)

NetSuite administrators who will be responsible for exposing third-party applications to NetSuite users should see [Making SuiteSignOn Integrations Available to Users](#). In cases where the administrator might also be responsible for building a custom integration should see the topics pertaining to application providers.



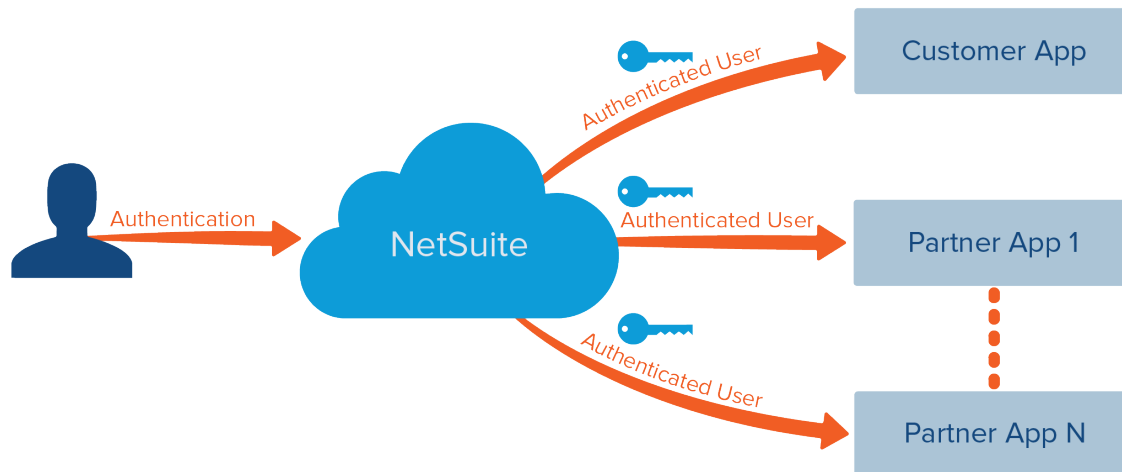
Important: Be aware of the following:

- NetSuite administrators should exercise caution when integrating with third party applications using SuiteSignOn. Some integrations may require access to, or even modify, some data in your NetSuite account. Make sure you review the data requirements and understand what kind of information is accessed, retrieved, modified, or deleted by the third party system. NetSuite has no control, responsibility, or liability regarding any third party applications, even if NetSuite offers resale and integration options for customers' convenience. You use and integrate with third party applications at your sole risk.
- Different features are available to implement inbound single sign-on from an external application to NetSuite. For information about these features, see:
 - [Inbound Single Sign-on](#)
 - [OpenID Single Sign-on](#)
 - [SAML Single Sign-on](#)

SuiteSignOn Overview

The outbound single sign-on implementation in NetSuite, called SuiteSignOn, enables users to be authenticated in the NetSuite user interface and move directly to an external user-authenticating Web application without additional authentication, from a link in the NetSuite user interface. These links, called connection points, currently are supported in NetSuite custom subtabs, custom portlets, Suitelets and user event scripts. NetSuite provides a SuiteSignOn setup page where application providers can enter data used for connection points.

Note: Calls initiated by Web Services are not supported by SuiteSignOn.



NetSuite authenticates each user upon login. When a user accesses a connection point, NetSuite initiates a two-way communication with the external application, and verification data passes between the two applications. This communication is referred to as a “handshake” because of its back and forth nature. After the handshake has been completed, the external application landing page displays in NetSuite. Also, if the application provider has included related web services calls in their application code, users' edits to external application data can be transferred to NetSuite.

Note: SuiteSignOn access from your web store is supported.

The benefits of the SuiteSignOn feature include improved usability, increased security and access control, reduced IT and support costs, use of NetSuite as the single trusted system for authentication, and simpler, more secure web services integration. For more details about these benefits and how to use SuiteSignOn, see [Outbound Single Sign-on \(SuiteSignOn\)](#). See also [Outbound Single Sign-on \(SuiteSignOn\) Access from Your Web Store](#).

Understanding SuiteSignOn

Each location in the NetSuite user interface where users can access an external application through SuiteSignOn is called a connection point. An application can have multiple connection points on different NetSuite pages. Currently, custom subtabs, custom portlets, and Suitelets are supported as connection points. User event scripts also are supported as connection points for web services integrations between NetSuite and external applications.

To implement single sign-on integration with an application, the application provider needs to set up information for each connection point. This information includes the name of the NetSuite subtab, portlet, Suitelet, or user event script connection point, the external application landing page, optional data that sets context for the landing page, and optional user identification data.

NetSuite authenticates each user upon login. When a user accesses a connection point, NetSuite initiates a two-way communication with the external application, and verification data passes between the two applications. This communication is referred to as a “handshake,” because of its back and forth nature. After the handshake has been completed, the external application landing page displays in the subtab, portlet, or Suitelet interface. Also, if the application provider has included related web services calls in their application code, users' edits to external application data can be transferred to NetSuite.

For more description of how SuiteSignOn connections work, see [SuiteSignOn Sequence Diagram and Connection Details](#).

For an overview of the ways in which a company can benefit from a SuiteSignOn implementation, see [SuiteSignOn Benefits](#).

SuiteSignOn Benefits

The SuiteSignOn feature provides the following benefits:

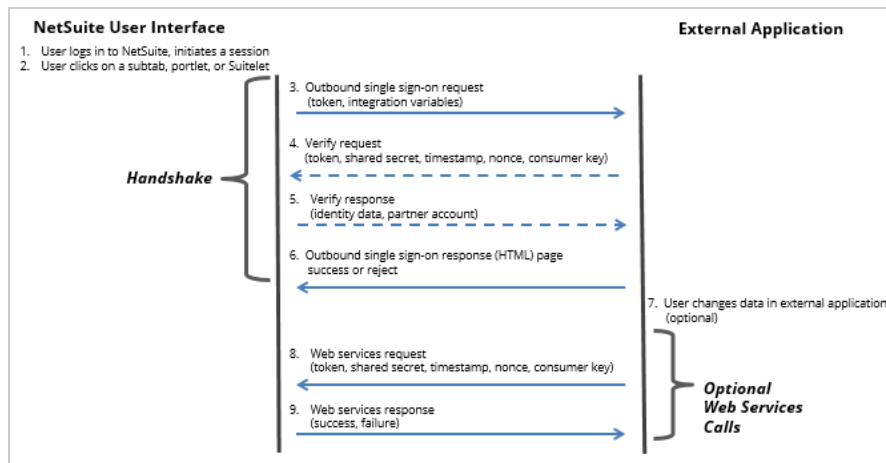
- Improved usability – Users can access other applications with their NetSuite login credentials, so they can complete daily tasks more quickly. They do not need to repeatedly log in and log out of multiple applications, or manage multiple sets of login credentials. They can log in a single time to NetSuite, and access an integrated solution within a single user interface - allowing users to complete daily tasks more quickly.
- Increased security and central access control – The password policy that is enforced for NetSuite access is enforced for any integrated application, providing consistency and limiting potential security issues.
- Reduced IT and support costs – The rollout of integrated applications is simplified because there is no need to maintain multiple databases for user credentials and access control.
- NetSuite remains the trusted system for authentication – Access from the NetSuite user interface to an external application user interface is confined to an iFrame. The external application does not have rights to change data in NetSuite except through specialized web services calls.
- Easier and more secure web services integration - The integrated application can use an already active session to transmit data to NetSuite via web services calls, instead of requiring the user to log in again. Changes submitted via web services are reflected in the NetSuite audit trail for the logged in user who makes the specific changes, instead of all changes being attributed to a single web services user. Web services uses the same role that was used to log in the user to NetSuite.

SuiteSignOn Sequence Diagram and Connection Details

The following sequence diagram illustrates the interaction between NetSuite and an external application during a SuiteSignOn connection.

- NetSuite is on the left and the external application is on the right.
- Steps 3-6 represent the handshake, meaning the calls required to verify the user and display the application in the NetSuite user interface.
- Steps 8-9 represent optional web services calls, used if the application provider wants to enable data transfer from the external application to NetSuite.

A detailed description of each step follows the sequence diagram.



SuiteSignOn Connection Details

See the following detailed steps for each action shown in the preceding SuiteSignOn connection sequence diagram.

- User logs in to NetSuite, initiating a NetSuite session.
- User clicks on one of the following in the NetSuite user interface:
 - A subtab that provides SuiteSignOn access
 - A page displaying a portlet that provides SuiteSignOn access
 - A link for a Suitelet that provides SuiteSignOn access
 - An action button that results in the execution of a user event script that provides SuiteSignOn access
- Outbound single sign-on request:** NetSuite generates a **token**, and sends this token to the external application as the value for the `oauth_token` URL parameter. This outbound HTTP call also includes a `dc` and an `env` URL parameter. These values can be mapped to the URL to be used for NetSuite access (see [Mappings of dc and env URL Parameter Values](#) and [Understanding NetSuite URLs and Data Centers](#)). If any data fields were previously defined as required context for the connection, NetSuite sends values for these fields at the same time.
- Verify request:** The external application sends back to NetSuite the token, the **consumer key**, and its **shared secret**, along with other information such as the timestamp and nonce, to verify the user.

The consumer key is a unique identifier for the application provider, generated by NetSuite when the application provider sets up a SuiteSignOn connection. The shared secret is a password defined by the application provider during this setup.
- Verify response:** NetSuite responds to the verification, sending any user identification information that was previously defined as necessary for the connection, in XML format. This information will be used by external application to uniquely identify the NetSuite user. For details about secure combinations of fields that should be used to uniquely identify users, please read [Choosing User Identification Fields for SuiteSignOn](#).
- Outbound single sign-on response:** The external application sends the HTML for the landing page, and the page displays. Or, if there is a problem, an error is returned instead.

Note: For user event script connection points, step 6 is omitted.

If the application provider has included web services calls for outbound single sign-on data transfer, the following actions may occur:

7. The user makes changes in the external application page displayed in NetSuite, then saves them.
8. **Web services request:** The external application sends a web services request, that includes the token and shared secret along with other verification data, to NetSuite.
9. **Web services response:** NetSuite sends a web services response to the external application, and either the changes are saved to NetSuite, or an error is returned.

Web services uses the same role that was used to log in to NetSuite.

Note the following:

- The token that NetSuite generates is good for the length of the UI session, or for 20 minutes of inactivity.
- If a user repeats step 2 multiple times during a single session, steps 4-5 can be skipped (at the discretion of the third-party client) after the first time.
- If the user logs out of NetSuite and logs back in, or switches roles, when the user clicks on the connection point, a new token is generated.

SuiteSignOn Required Features

Application providers and NetSuite administrators may need to enable certain features to facilitate a SuiteSignOn implementation. The following table lists each feature and whether it must be enabled, depending on the scenario.

All of the features below can be enabled at Setup > Company > Enable Features, on the SuiteCloud subtab.

Feature	Required for Application Providers?	Required for SuiteSignOn User Accounts?	Notes
SuiteSignOn	Yes	Yes	
SuiteTalk (Web Services)	Maybe	Maybe	Required if SuiteSignOn code includes web services calls.
SuiteBundler	Yes	No	Required to create and deploy bundles. Not required to install bundles.
Client and Server SuiteScript	Maybe	Maybe	Required if the connection points are created using custom portlets, Suitelets, or user event scripts, client and server SuiteScript should be enabled. Not required for subtab connection points.

Setting Up SuiteSignOn Integration

The following tasks can be completed by anyone who has the SuiteSignOn permission. However, most of these tasks will be completed by application providers wanting to implement a SuiteSignOn integration with NetSuite.

If you are a NetSuite administrator who is working with an application provider to build a custom solution (or you are managing a bundled solution), you may also end up performing some of these tasks. Typically, however, most administrators will complete the tasks outlined in [Making SuiteSignOn Integrations Available to Users](#).

Summary of integration tasks:

1. Enable the SuiteSignOn feature and other required SuiteCloud features in your NetSuite account. See [SuiteSignOn Required Features](#).
2. Application providers must add required code to their application to support the exchange of token and shared secret information with NetSuite, referred to as the “handshake”. For sample code, see [SuiteSignOn Code Samples](#). These code additions include:
 - a verify call in the HTTP header and code that requests token verification from NetSuite
 - (optional) web services calls to transfer data between NetSuite and your application
3. Create one or more custom subtabs, portlets, Suitelets or user event scripts to be connection points that provide access to the integrated application. See [Creating SuiteSignOn Connection Points](#).



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

4. (Optional) Define any custom entity fields as user identification fields. Ensure that these fields have been created and marked as Available to SuiteSignOn. Also, you need to determine a way for account administrators to enter or import values for these fields as needed. See [Using Custom Fields as SuiteSignOn User Identification](#).
5. Create a NetSuite SuiteSignOn record. See [Creating SuiteSignOn Records](#).
6. (Application providers) Create a SuiteBundle that includes SuiteSignOn connection data and custom objects, write bundle documentation instructing administrators how to set it up in their accounts, and make the bundle available to NetSuite users. See [Creating a SuiteSignOn Bundle](#).
7. (NetSuite administrator) Install the SuiteSignOn bundle created by the application provider. See [Making SuiteSignOn Integrations Available to Users](#).

Creating SuiteSignOn Connection Points

A connection point is the place in the NetSuite user interface where users access the external application. A single application can have multiple connection points on different NetSuite pages.

Currently, custom subtabs, custom portlets, Suitelets, and user event scripts are supported as connection points.



Note: Calls initiated by Web Services are not supported by SuiteSignOn.

The type of connection point you create depends on how you want NetSuite users to access the integrated application. See [Comparing Subtab, Portlet, Suitelet and User Event Connection Points](#) to help you decide which type of connection point best suits your implementation.



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

To create one of the supported connection points, see these topics:

- [Creating a Custom Subtab Connection Point](#)
- [Creating a Portlet Connection Point](#)
- [Creating a Suitelet Connection Point](#)
- [Creating a User Event Connection Point](#)

Creating a Custom Subtab Connection Point

You can create custom subtabs to provide outbound single sign-on access within NetSuite records. The following types of custom subtabs are available:

- **Transaction** — Can be displayed on transaction records such as sales order, cash sale, opportunity
- **Entity** — Can be displayed on entity records such as customer, vendor, employee
- **Item** — Can be displayed on item records such as inventory, non-inventory, assembly/bill of materials
- **CRM** — Can be displayed on CRM records such as task, phone call, event
- **Custom Record Subtab** — Can be displayed on its associated custom record



Note: For a complete list of transaction, entity, item, and CRM records that support custom subtabs, see the help topic [Creating Custom Subtabs](#) in the NetSuite Help Center.

To create custom subtabs:

1. Go to Customization > Forms > Subtabs, and follow step a or step b:
 - a. Click the subtab for the type of record where you want to create a new subtab: **Transaction, Entity, Item, or CRM**,
OR:
 - b. Edit a custom record type record, either by going to Customization > Lists, Records, & Fields > Record Types and clicking an **Edit** column link, or by going to Customization > Lists, Records, & Fields > Record Types > New, and click the **Subtabs** subtab.
2. Enter the name for your subtab in the **Title** field. NetSuite recommends using a name that is a meaningful reference to your application, because it serves as the subtab label in the NetSuite user interface.
3. If desired, designate this subtab as a child of an existing subtab. In the **Parent** field, select an existing subtab from the list.
4. Click **Add**.
5. Repeat these steps for each subtab you want to create.
6. Click **Save**.

After you have created a custom subtab, you can define it as a connection point on the SuiteSignOn page. For each subtab connection point, you can specify a single record type to which it applies. The subtab is displayed on that record type's forms. When the record that includes the subtab is loaded, the SuiteSignOn connection is initiated, resulting in the display of an iFrame rendering your application.

Note the following:

- Usually, you need to add at least one custom field to a custom subtab for it to display on forms. However, subtabs that are defined as SuiteSignOn connection points do not require any custom fields. In fact, NetSuite recommends that you do not add any fields to these subtabs.

- You can control the records to which a subtab is applied when you set up subtab connection points on the SuiteSignOn page. See [Creating SuiteSignOn Records](#).
- You can limit the users who have access to each record type subtab by customizing the record type form and setting up preferred forms for users.
- For a description of the differences between different types of connection points, see [Comparing Subtab, Portlet, Suitelet and User Event Connection Points](#).

Creating a Portlet Connection Point

You can create portlet scripts to provide outbound single sign-on access in custom portlets. To make a portlet script available for a connection point, you need to create a JavaScript file, create a NetSuite script record, and deploy the script.

For more information, see the following topics in the NetSuite Help Center:

- [Portlet Scripts](#)
- [Running Scripts in NetSuite Overview](#)
- [nlapiOutboundSSO\(id\)](#)

To create and deploy a SuiteSignOn portlet script:

1. Create a .js file that uses the `nlapiOutboundSSO(id)` API, with an argument of either: the ID specified on the SuiteSignOn record, or the SuiteSignOn record's internal ID.

NetSuite recommends that you use the first option. For information about specifying this value on the SuiteSignOn record, see [Setting SuiteSignOn Basic Definitions](#).

The following is a sample portlet script that displays a SuiteSignOn enabled external application in the NetSuite user interface:

```
function buildPortlet(portlet, column)
{
    portlet.setTitle('My Integrated Application!!');
    var url = nlapiOutboundSSO('customsso_myApp');
    var content = '<iframe src="'+url+'" align="center" style="width: 100%; height: 600px; margin:0; border:0; padding:0"></iframe>';
    portlet.setHtml( content );
}
```

2. Create a record for the script and deploy it in NetSuite.
 - a. Go to Customization > Scripting > Scripts > New.
 - b. On the Select Type page, choose **Portlet**.
 - c. On the script page, enter a name and a script ID, and choose a **Portlet** Type.



Note: The **Inline HTML** portlet type is probably the easiest to implement.

NetSuite recommends that you specify a unique script ID if you intend to bundle and distribute your SuiteSignOn integration.

- d. On the **Scripts** tab, enter the name of the script file created in step 1, and the function used, **buildPortlet**. (Do not type the function parentheses or any function arguments.)
- e. Save the script record, then click **Deploy Script**.

- f. On the Script Deployment page, be sure to define an audience for the portlet script. The audience definition lets you limit access to the portlet.
 - g. Save the script deployment record.
3. Define a portlet connection point for your SuiteSignOn integration. See [Defining SuiteSignOn Connection Points](#). See [Adding Custom Portlets for SuiteSignOn](#) to learn how to make custom SuiteSignOn portlets available to users.

Notes:

- For more general information on portlet scripts, see the help topic [Portlet Scripts](#) in the NetSuite Help Center.
- To learn more about defining integration variables for portlet connection points, see [Defining Integration Variables for Connection Points](#).
- For a description of the differences between different types of connection points, see [Comparing Subtab, Portlet, Suitelet and User Event Connection Points](#).

Creating a Suitelet Connection Point

You can create Suitelets to provide outbound single sign-on access in custom user interface objects. To make a Suitelet available for a SuiteSignOn connection point, you need to create a JavaScript file, create a NetSuite script record, and deploy the script.

For more information, see the following topics in the NetSuite Help Center:

- [Suitelets](#)
- [Running Scripts in NetSuite Overview](#)
- [nlapiOutboundSSO\(id\)](#)



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

To create and deploy a SuiteSignOn Suitelet:

1. Create a .js file that uses the `nlapiOutboundSSO(id)` API, with an argument of either: the ID specified on the SuiteSignOn page, or the SuiteSignOn record's internal ID.

NetSuite recommends that you use the first option. For information about specifying this value on the SuiteSignOn page, see [Setting SuiteSignOn Basic Definitions](#).

The following is a sample Suitelet script:

```
function buildSuitelet(request, response)
{
    if ( request.getMethod() == 'GET' )
    {
        //Create a form.
        var form = nlapiCreateForm('Test Outbound SSO Suitelet');

        var label = form.addField('custpage_label', 'inlinehtml', 'SSO');
        label.setDefaultValue ('<B>Check out my SSO Suitelet!!</B>');
```

```

        var url = nlapiOutboundSSO('customsso_myApp');
        var content = '<iframe src="'+url+'" align="center" style="width: 1000p
x; height:
        800px; margin:0; border:0; padding:0"></iframe>';

        var iFrame = form.addField('custpage_sso', 'inlinehtml', 'SSO');
        iFrame.setDefaultvalue (content);
        iFrame.setLayoutType('outsidebelow', 'startcol');
        response.writePage( form );
    }
}

```

2. Create a record for the script and deploy it in NetSuite.
 - a. Go to Customization > Scripting > Scripts > New.
 - b. On the **Select Type** page, choose **Suitelet**.
 - c. On the script page, enter a name and a script ID.
 NetSuite recommends that you specify a unique script ID if you intend to bundle and distribute your SuiteSignOn integration.
 - d. On the **Scripts** tab, enter the name of the script file created in step 1, and your Suitelet script's main function **buildSuitelet**. (Do not type the function parentheses or any function arguments.)
 - e. Save the script record, then click **Deploy Script**.
 - f. On the Script Deployment page, do the following:
 - On the **Audience** subtab, define an audience for the Suitelet. The audience definition lets you limit access to the Suitelet.
 - On the **Links** subtab, define tasklinks to provide access to the Suitelet. Select one or more centers to include Suitelet access, and for each center, define the tab (section) to include the tasklink and a label for the tasklink.
 - g. Save the script deployment record.
3. Define a Suitelet connection point for your SuiteSignOn integration. See [Defining SuiteSignOn Connection Points](#).

Notes:

- For more general information on working with Suitelets, see the help topic [Suitelets](#) in the NetSuite Help Center.
- To learn more about defining integration variables for Suitelet connection points, see [Defining Integration Variables for Connection Points](#).
- For a description of the differences between different types of connection points, see [Comparing Subtab, Portlet, Suitelet and User Event Connection Points](#).

Creating a User Event Connection Point

You can create user event scripts that use SuiteSignOn to support real-time integration between NetSuite and external applications. User event scripts execute at one of the following points: when a read operation on a record takes place (Before Load), when a record is submitted before changes are committed to the database (Before Submit), or when changes are committed to the database (After Submit). Through SuiteSignOn, a user event script can notify an external application of record updates, passing each record ID as a URL parameter. The external application can then access NetSuite through web services calls, to acquire additional information about these records.



Important: The external system's access to NetSuite is limited to the access available for the user who performed the action that caused user event script execution.

To make a user event script available for a SuiteSignOn connection point, you need to create a JavaScript file, create a NetSuite script record, and deploy the script.

To set up a SuiteSignOn user event script:

1. Create a .js file that uses the `nlapiOutboundSSO(id)` API, with an argument of either: the ID specified on the SuiteSignOn page, or the SuiteSignOn record's internal ID.
 - NetSuite recommends that you use ID specified on the SuiteSignOn page as an argument. For information about setting this value, see [Setting SuiteSignOn Basic Definitions](#).
 - For details about the API, see the SuiteScript help topic [nlapiOutboundSSO\(id\)](#).
 - For details about creating user event script files, see the SuiteScript help topic [User Event Scripts](#).

The following is a sample user event script:

```
function syncWithExternalApp(type)
{
    var url = nlapiOutboundSSO('customsso_my_external_app');
    nlapiRequestURL(url);
}
```

2. Create a record for the script and deploy it in NetSuite.

On the script record, you indicate when the script is executed by the field in which you enter the function name: **Before Load**, **Before Submit**, or **After Submit**.

For details about creating script records and deploying scripts, see the SuiteScript help topic [Running Scripts in NetSuite Overview](#).

3. Define a SuiteSignOn record for the user event connection point at Setup > Integration > SuiteSignOn > New. For details on these steps, see [Creating SuiteSignOn Records](#) and [Defining SuiteSignOn Connection Points](#).



Note: To learn more about defining integration variables for user event connection points, see [Defining Integration Variables for Connection Points](#).

Comparing Subtab, Portlet, Suitelet and User Event Connection Points

If you are trying to decide whether to set up subtab, portlet, Suitelet, or user event connection points for your application, consider the following: how comfortable you are with scripting, where the application should display within the NetSuite user interface, and how you want it to look.


- Subtab connection points may be simplest to implement, because they do not require scripting.
- A portlet connection point provides greater flexibility than a subtab in how the application looks; a Suitelet provides even more flexibility.
- A Suitelet is the only connection point supported for SuiteSignOn access from your web store.
- A user event connection point provides integration with an external application without exposing it in the NetSuite user interface.

The following table outlines differences:

Functionality	Subtab	Portlet	Suitelet	User Event
Required NetSuite customizations	Creation of a custom subtab; no scripting required.	Custom scripting required; <code>nlapiOutboundSSO(id)</code> .	Custom scripting required; <code>nlapiOutboundSSO(id)</code> .	Custom scripting required; <code>nlapiOutboundSSO(id)</code> .
Availability on dashboard	Visible within specifically defined records. Automatically available after bundle installed.	Can be added to any page that allows custom portlets. Not available until custom portlet is added and configured to display script.	Link can be added to any page menu. Automatically available after bundle installed.	Not exposed in user interface. Connection is initiated either Before Load, Before Submit, or After Submit, based on the user event script record function.
Ability to modify application "look and feel"	Very limited; external application landing page displayed within iFrame is only subtab contents.	Based on script, so can be free-form.	Based on script, so can be free-form.	Not exposed in user interface.
Required connection point definitions	Need to define separate connection point for each subtab integration.	One connection point can provide integration in multiple portlets.	Need to define separate connection point for each Suitelet integration.	Need to define separate connection point for each integration.

Creating SuiteSignOn Records

You need to create one SuiteSignOn record for each application that you want to integrate with NetSuite. Each application may have multiple connection points, all of which are listed on the same record.


 **Important:** You must have the SuiteSignOn permission to create or edit SuiteSignOn records.

- To create a new SuiteSignOn record for an application, go to Setup > Integration > SuiteSignOn > New.
- To edit an existing SuiteSignOn record for an application, go to Setup > Integration > SuiteSignOn and click the Edit link for a record.
See [Editing SuiteSignOn Records](#).

On the SuiteSignOn page, you can complete the following tasks:

- [Setting SuiteSignOn Basic Definitions](#)
- [Defining SuiteSignOn Connection Points](#)
- [Choosing User Identification Fields for SuiteSignOn](#) (optional)
- [Using Custom Fields as SuiteSignOn User Identification](#) (optional)
- [Dynamically Mapping User Identification Information](#) (optional)

After you are done with these tasks, you can build a SuiteSignOn bundle to distribute to your customers. See [Creating a SuiteSignOn Bundle](#).

 **Warning:** For bundled SuiteSignOn integrations, the SuiteSignOn record is completed by the application provider, and administrators install the bundle that contains this data, making edits to the record as instructed in the bundle document. If administrators attempt to make other edits to this page, issues are likely to arise. See [Making SuiteSignOn Integrations Available to Users](#).

Setting SuiteSignOn Basic Definitions


On the SuiteSignOn record for an external application, you (the application provider) need to define the following:

- **Name** - A name for the external application integration, to display in NetSuite lists.
- **ID** - A script ID for this integration, to be passed as a parameter in the portlet script. The value you enter here is automatically prepended with **customsso_**. NetSuite recommends that you assign a unique script ID to your SuiteSignOn object if you intend to bundle and distribute your integration.
- **Shared Secret** - A password used to establish ownership of the Consumer Key generated by NetSuite. This value is included as the signature passed in your HTTP header, and needs to be referenced in your application verification code.

The shared secret must comply with the requirements specified in RFC 5849 - OAuth 1.0, sections 3.4.4, 3.5.1 and 3.6. The OAuth signature must include the & character (an ampersand, ASCII code 38) and the space character (+) must be encoded as %20. When double-encoded, the space character %20 becomes %2520.

For SuiteSignOn, the format is: `signature = urlencode(urlencode(shared secret) + '&')`

For example, if you chose P@mpired15! as your shared secret, when encoded, the signature would be: "P%2540mpired15%2521%26"

 **Important:** See [Notes about Modifying the Shared Secret](#) for tips about changes to this password.


You do not need to define the following:

- **Consumer Key** - You cannot enter or edit this globally unique identifier for your application. It is generated by NetSuite. You will need to include this value in your HTTP header and application verification code.
- **Partner Account** - Each customer's account ID for your application. Each customer may need to enter this value after installation of the SuiteSignOn bundle. This value is not necessary if your integrated application does not require this value for identification. Be sure to include instructions for this task in your bundle documentation, if necessary.
- **Web Services Access** - Level of access supported for web services callbacks from integrated applications. The following options are available:
 - Same as UI Role - the default, which allows web services callbacks from integrated applications with the same level of permissions as in the user interface integration.
 - No Access - prevents integrated applications from accessing NetSuite through web services callbacks.
 - Additional options for any web services only roles in the account - selecting one of these roles allows web services callbacks from integrated applications, but limits access to the permissions levels assigned to the selected role.

As a security best practice, you should provide the minimum level of access required for SuiteSignOn integrated applications. For example, if an application only requires user interface integration, it is best to set the Web Services Access option to No Access.

The Web Services Access field is also available for viewing and editing on the SuiteSignOn list page at Setup > Integration > SuiteSignOn.

After you have set basic definitions for the SuiteSignOn integration, you can define one or more connection points where your application is displayed in the NetSuite user interface, and user identification fields that are used as context for the integration.

 **Note:** For examples of HTTP header and application verification code, see [SuiteSignOn Code Samples](#).

Defining SuiteSignOn Connection Points

Connection points are the locations in the NetSuite user interface that provide access to your application. Every application integrated through SuiteSignOn can have multiple connection points.

To set up connection points for your application, define the following on the SuiteSignOn page:


- URL - Enter the URL for the external application landing page to be displayed in the connection point. This page must be secure, with an https:// URL. SuiteSignOn is not supported for http:// sites. You can specify a different URL for each connection point.
- Integration Variables - You can optionally define one or more field values to be passed as context in the initial HTTP call from NetSuite to your application, before authentication. For an example of this call, see [NetSuite HTTP Outbound Call](#).
 - You do not need to specify integration variables if context is included in the URL.
 - Both static and dynamic field values are supported.
 - To specify a static value, use a format like **q=value**, as in the following examples:
`customer_id=12345`
`first=John`
`last=Smith`
`mid=Jay`
 - To specify a dynamic value, use a format like **q={field_id}**, as in the following examples:
`customer_id={id}`
`first={firstname}`
`last={lastname}`
`mid={middlename}`
 - To specify a null value, use a format like **q=**
 - Variables can include spaces. Static variables cannot include commas within values.
 - You can use comma-separated values or carriage return-separated values to specify multiple values.
 - You cannot use social security numbers, passwords, or credit card numbers as integration variables, because of the potential security risks. If you do so, they will not be returned.
 - For subtab connection points, standard and custom fields on the form for the specified record type can be used as integration variables. You cannot use fields that are not included on the form. If a referenced field has no value, no value is passed as an integration variable.
 - For portlet, Suitelet, and user event connection points, see [Defining Integration Variables for Connection Points](#). This section provides detailed information regarding the use of static and dynamic integration variables in these connection points.

- Display Type - Choose Subtab, Portlet, Suitelet, or User Event.
- Display Context - Choose the name of the subtab, portlet, Suitelet, or user event script to be used for SuiteSignOn access.

A subtab, portlet script, Suitelet, or user event script must already exist in your account to be listed in this dropdown. See [Creating a Custom Subtab Connection Point](#), [Creating a Portlet Connection Point](#), [Creating a Suitelet Connection Point](#), and [Creating a User Event Connection Point](#).

- Record Type - (Subtabs only) Choose the record type for each subtab connection point. Custom record types are available for their associated subtabs.

If you want to display the external application in a custom subtab on multiple record types (for example, on contacts, customers, and partners), you need to add multiple connection points, with the same Display Type and Display Context, and a different Record Type for each.

 **Note:** Be sure to click Add after you enter each connection point.

Defining Integration Variables for Connection Points

You can define both static and dynamic integration variables for portlet, Suitelet, and user event connection points. Note, however, if you want to define dynamic integration variables for either of these connection types, you must do so in the portlet, Suitelet, or user event JavaScript file. You cannot define dynamic integration variables in the Integration Variables field on the SuiteSignOn record.

The following code shows a portlet script. Notice that the script calls `nlapiGetContext()`, which returns a `nlobjContext` object. Through this object, you can obtain information about the currently logged-in user. In this script, the `nlobjContext.getEmail()` method is used to get the email address of the currently logged-in NetSuite user. The value of the email address can then be passed to the URL as a dynamic integration variable.

```
function buildPortlet(portlet, column)
{
    title = 'My Integrated Application!!'
    portlet.setTitle(title)

    // return the nlobjContext object
    var ctx = nlapiGetContext();

    // use getEmail() to obtain the logged-in user's email address
    var email = ctx.getEmail();

    var url = nlapiOutboundSSO('customsso_myApp');

    // set the integration variables to be passed in the URL. In this case, a static value is set for the
    // 'partner' variable. The value of the 'email' parameter will be the value of the email variable.
    url = url + '&partner=NetSuite' + '&email=' + email;

    var content = '<iframe src="'+url+'" align="center" style="width: 100%; height: 600px; margin:0;border:0;padding:0"></iframe>';

    portlet.setHtml( content );
}
```

Note: The following screenshot shows that you could have used the Integration Variables field on the SuiteSignOn record to define the **partner** integration variable, which is static. You could not have used this field to define a dynamic integration variable, such as the **email** variable in the script above.

The screenshot shows the 'User Identification' tab of a SuiteSignOn record. The 'URL' field contains 'https://www.website.com'. The 'INTEGRATION VARIABLES' field is highlighted with a red box and contains the value 'partner=NetSuite'. The 'DISPLAY TYPE' is set to 'Portlet'.

Choosing User Identification Fields for SuiteSignOn

The user identification fields are used by external applications to uniquely identify the NetSuite user. User identification fields are defined per application, so they are the same for each connection point. These fields' values are passed in XML format in the HTTP response from NetSuite to your application after verification. For an example, see [NetSuite HTTP Verify Call Response](#).

The following user identification fields are provided on the SuiteSignOn page:

- Email - Email address used as the user ID for NetSuite
- Account - Customer's NetSuite account ID
- First Name
- Middle Name
- Last Name
- Internal ID - NetSuite-generated unique identifier
- External ID - External application unique identifier stored in NetSuite

To uniquely identify each user across all NetSuite accounts, you should use one of the following two combinations of data:

- The customer's NetSuite account ID and the user's email address.
- The customer's NetSuite account ID and the user's internal ID.

You also can make custom entity fields available on this page, by checking the Available to SuiteSignOn box on the Custom Entity Field record. See [Using Custom Fields as SuiteSignOn User Identification](#).

The screenshot shows the 'User Identification' tab of a SuiteSignOn record. It displays a list of fields with checkboxes: Email (checked), Account, First Name, Middle Name, Last Name, Internal ID, External ID, SSO Pwd (checked), and SSO username (checked). At the bottom, there are 'Save' and 'Cancel' buttons.

Using Custom Fields as SuiteSignOn User Identification

In addition to the standard fields available when you are [Choosing User Identification Fields for SuiteSignOn](#), you can define entity custom fields to be available for this purpose. NetSuite passes values of the user identification fields selected on the SuiteSignOn page to your application.



Important: In addition to any custom fields that you choose to include, every user should be identified by a unique combination of data defined in standard NetSuite fields. For details on the two combinations that are supported, see [Choosing User Identification Fields for SuiteSignOn](#).

To make a custom entity field available for user identification:

- If the field does not yet exist in NetSuite, add it at Customization > Lists, Records, & Fields > Entity Fields.
- Ensure that the field is available on all types of records corresponding to the users who can access your SuiteSignOn integration. These may include employees, customers, partners, and vendors.
- Check the Available to SuiteSignOn box. After this box is checked, the custom field is listed on the User Identification subtab of the SuiteSignOn page.

For instructions for setting up custom fields, see the help topic [Creating a Custom Field](#).

You need to determine a way for account administrators to populate custom field values, either through manual entry, mass update, CSV import, or another import process. NetSuite recommends that you provide instructions for this task in your SuiteSignOn bundle documentation. If you want values to be populated through CSV import, you can save an import map and include it in the bundle. See the help topics [Working with Saved CSV Imports](#) and [Creating a SuiteSignOn Bundle](#).

Dynamically Mapping User Identification Information

If the default identity information return by standard SuiteSignOn ID fields is insufficient for a certain application, you can establish dynamic identity mappings. To establish dynamic identity mappings, you can design a landing page and prompt each user upon first connection to log in, or enter their ID in the third-party system, and submit the mapping back into a custom entity field, which is available to SuiteSignOn through web services.

After the field is populated, subsequent connections into the third-party application will bypass the initial landing page and the identity of the user is known. For this approach to work, the role of the logged-in user in NetSuite should have permission to update their own entity record to set the custom field.

If the logged in user's role does not have permission to update the entity record, a custom record can be created to track identity mappings for a certain SuiteSignOn integration. Another entity custom field that is available to SuiteSignOn can source the mapping from the custom record.

For instructions for setting up custom fields, in the NetSuite Help Center see the help topic [Creating a Custom Field](#). For instruction on creating custom records, see the help topic [Custom Records](#).

Editing SuiteSignOn Records

After a SuiteSignOn record has been created, users with the SuiteSignOn permission can edit this record as needed.

If you make changes to a SuiteSignOn record after it has been bundled and distributed, you must update the bundle in the repository if it was copied there, and inform bundle users of the change, so they can update their installations to get the latest version. See the help topic [Using the Bundle Repository](#).

To edit a SuiteSignOn record:

1. Go to Setup > Integration > SuiteSignOn, and click the **Edit** link for a record.

2. Make changes as desired. For details about definitions that can be edited, see:

- [Setting SuiteSignOn Basic Definitions](#)
- [Defining SuiteSignOn Connection Points](#)
- [Choosing User Identification Fields for SuiteSignOn](#)
- [Using Custom Fields as SuiteSignOn User Identification](#)

Notes about Modifying the Shared Secret

- You cannot change the Shared Secret value unless you are the creator of the SuiteSignOn record.
- If you change the Shared Secret after your SuiteSignOn solution has been installed in other accounts, you cause this password to change for all instances of the SuiteSignOn integration across all accounts in both the production and sandbox domains. So, for example, if you modify the Shared Secret on a SuiteSignOn record in a sandbox account, it is changed in production accounts as well.
- See [Setting SuiteSignOn Basic Definitions](#) for more information about requirements for the Shared Secret.

Disabling a SuiteSignOn Integration

You can mark a SuiteSignOn integration as inactive either on the record itself, by checking the Inactive box, or on the SuiteSignOn list page, by checking the Show Inactives box, then checking the Inactive box for the record. When a SuiteSignOn record is inactive, any subtab connection points are not displayed, and portlet scripts and Suitelets return errors.

Creating a SuiteSignOn Bundle

After you have completed the tasks in [Setting Up SuiteSignOn Integration](#), you can use SuiteBundler to package your SuiteSignOn objects for distribution.

The following table lists common SuiteSignOn bundle objects:

Object	When to include in SuiteSignOn Bundle
SuiteSignOn Outbound Connection	Always Any custom subtabs, portlet scripts, and Suitelets defined as connection points, and any custom fields defined as user identification, are automatically included with the SuiteSignOn Outbound Connection object.
Custom Field(s)	If integration uses custom fields as integration variables Custom fields defined as user identification are automatically added.
Saved CSV Import	If integration uses custom fields as integration variables or user identification, and you want to provide a predefined import mapping for populating these fields' values

You also should include bundle documentation, a file that provides instructions for account administrators who install the bundle.

SuiteSignOn bundles are customization bundles, not configuration bundles. For more information, see the help topic [SuiteApp Creation and Distribution](#).

To bundle your SuiteSignOn integration:

1. Create your bundle documentation file. This file should include a description of the bundle contents and a list of steps that are required after bundle installation.

- To help administrators verify SuiteSignOn page contents, you may want to include a checklist of values for the basic information fields, connection point details, and boxes that should be checked on the **User Identification** subtab. Or, you could include a screenshot of this page in the bundle documentation file.
 - For details about other steps you may need to explain in this file, see [Making SuiteSignOn Integrations Available to Users](#).
2. Go to Customization > SuiteBundler > Create Bundle to start the Bundle Builder, and follow the instructions in the SuiteBundler help topic [Creating a Bundle with the Bundle Builder](#).
 3. To enable administrators to install your bundle, communicate to them the bundle name and ID. Also let them know whether they should install the bundle from the bundle repository, from your production account, or from your sandbox account, and provide the account ID as necessary.

Note: The Web Services Access option in a bundled SuiteSignOn record is pushed to target accounts as part of new bundle installations. However, a change to this option is not pushed to target accounts during bundle updates, to prevent overwriting account administrators' choices.

Making SuiteSignOn Integrations Available to Users

NetSuite administrators can enable SuiteSignOn integrations in their account by completing the following tasks. If you are not familiar with the SuiteSignOn feature, NetSuite recommends that you see [Outbound Single Sign-on \(SuiteSignOn\)](#) and [Understanding SuiteSignOn](#).

Warning: NetSuite administrators should exercise caution when integrating with third party applications using SuiteSignOn. Some integrations may require access to, or even modify, some data in your NetSuite account. Make sure you review the data requirements and understand what kind of information is accessed, retrieved, modified, or deleted by the third party system. NetSuite has no control, responsibility, or liability regarding any third party applications, even if NetSuite offers resale and integration options for customers' convenience. You use and integrate with third party applications at your sole risk.

Summary of tasks:

1. Enable SuiteSignOn-related features (see [SuiteSignOn Required Features](#)).
2. Install a SuiteSignOn bundle (see [Installing a SuiteSignOn Bundle](#)).
3. Complete the implementation tasks required for making the third-party application available to NetSuite users (see [Completing Account Setup for SuiteSignOn](#)).

Note: The tasks mentioned here are aimed at account administrators. If you are an application provider and want to create a SuiteSignOn integration, see [Setting Up SuiteSignOn Integration](#).

Installing a SuiteSignOn Bundle

The main requirement to implement SuiteSignOn integration in your account is to install a bundle. The application provider that created the bundle should let you know the bundle name and ID. Also, they should indicate where you can find the bundle, either in the Bundle Repository or in an account. If you need to install a bundle directly from an account, you will need the account ID.

To install a SuiteSignOn bundle:

1. Go to Customization > SuiteBundler > Search & Install Bundles, and follow the instructions in [Installing a Bundle](#).
2. Follow the instructions in the bundle documentation file to completely implement SuiteSignOn in your account. See [Completing Account Setup for SuiteSignOn](#).

Completing Account Setup for SuiteSignOn

After you have finished [Installing a SuiteSignOn Bundle](#) in your account, you need to complete a few additional setup tasks in NetSuite.

The bundle documentation file should include instructions for these tasks. If you have not yet reviewed this file, go to Customization > SuiteBundler > Search & Install Bundles > List, and on the Installed Bundles page, click the Documentation link.

You should follow the instructions in this file. The list below describes tasks that are likely to be included in this file:

1. Go to Customization > SuiteBundler > SuiteSignOn, click the link for the newly installed SuiteSignOn integration, and verify that the SuiteSignOn page looks correct. The bundle documentation should include a checklist or a screenshot for you to use for this purpose.
2. Make changes to the SuiteSignOn page as necessary. You must have the SuiteSignOn permission to edit SuiteSignOn records.
 - If your account ID for the application provider is required for identification, enter it in the Partner Account field. This value is not necessary if the integrated application does not use it for identification. The bundle documentation should indicate whether this value is required.
 - If you want to control the level of NetSuite access for web services callbacks from integrated applications, change the Web Services Access option. The following options are available:
 - Same as UI Role - the default, which allows web services callbacks from integrated applications with the same level of permissions as in the user interface integration.
 - No Access - prevents integrated applications from accessing NetSuite through web services callbacks.
 - Additional options for any web services only roles in the account - selecting one of these roles allows web services callbacks from integrated applications, but limits access to the permissions levels assigned to the selected role.

As a security best practice, you should provide the minimum level of access required for SuiteSignOn integrated applications. For example, if an application only requires user interface integration, it is best to set the Web Services Access option to No Access.

This field is also available for viewing and editing on the SuiteSignOn list page at Setup > Integration > SuiteSignOn.




Important: Be aware that changing the Web Services Access option could possibly break an integration, because some integrations may depend on existing user permissions.

- If the bundle includes custom fields to be used as user identification, ensure that they appear on the User Identification subtab and are checked. The bundle documentation should indicate whether these fields are included.

Be aware that the names of bundled custom fields may be changed slightly from those listed in the bundle documentation, if any of their IDs conflict with preexisting custom fields in

your account. If a conflict is detected, the bundled custom field ID is appended with “_#”. For example, if a bundle installs a custom field with an ID of custentitybanana and a preexisting custom field has the same ID, the bundled field ID is changed to custentitybanana_2. This field ID also is changed where it is referenced in SuiteSignOn setup information, either in the Integration Variables field, or on the User Identification subtab.

 **Warning:** NetSuite administrators should exercise caution when integrating with third party applications using SuiteSignOn. Some integrations may require access to, or even modify, some data in your NetSuite account. Make sure you review the data requirements and understand what kind of information is accessed, retrieved, modified, or deleted by the third party system. NetSuite has no control, responsibility, or liability regarding any third party applications, even if NetSuite offers resale and integration options for customers' convenience. You use and integrate with third party applications at your sole risk.

3. You may need to populate values for custom fields used for user identification, through manual entry, mass update, CSV import, or another import process. Follow the bundle documentation instructions for this task.
4. If any portlet connection points are included, you will need to add one or more custom portlets that display the specified scripts to your dashboard and publish it to other users, or you will need to provide instructions to users for adding custom portlets to their own dashboards. See [Adding Custom Portlets for SuiteSignOn](#).
5. If you do not want a subtab connection point to be available to all users with access to the specified record type, you can create a custom form that hides the subtab, define this custom form as preferred for some users, and restrict their access to other forms for that record type. See the help topics [Creating Custom Entry and Transaction Forms](#) and [Defining Preferred Entry and Transaction Forms](#).

Adding Custom Portlets for SuiteSignOn

After you have installed a SuiteSignOn bundle, any subtab, Suitelet, and user event connection points are immediately available to users in your account. However, a portlet connection point is not available to users until they have added a custom portlet to the dashboard and configured that portlet to use the script defined for that connection point.

You can do either of the following to expose a portlet connection point:

- Add a custom portlet to your own dashboard and publish it to users. This option provides you with greater control.
See the help topic [Publishing Dashboards Overview](#) . Be aware that you can only publish a dashboard to users with the same center as you, so you may need to log in with multiple roles and repeatedly add the custom portlet to multiple dashboards to make the portlet available to users with different centers.
- Provide users with instructions for adding a custom portlet to their dashboards.

To expose a portlet connection point on your dashboard:

1. On the page where you want to add the connection point, click the **Customize this Page** link.
2. In the **Add Content** panel, drag and drop a **Custom Portlet** object to the desired location on the page.
3. In the **Custom Content** portlet, click the **Set Up** link.
4. In the **Set Up Scripted Content** dialog, select the name of the script that is listed as the portlet connection point, and click **Save**.

SuiteSignOn Code Samples

The SuiteSignOn feature uses a portion of the OAuth protocol specification. OAuth enables applications to access another application's protected resources from a Web service via an API, without requiring users to disclose to the first application their credentials for the second application. The OAuth specification is available at <http://oauth.net/documentation/spec>.

To understand how the SuiteSignOn feature uses OAuth terminology, see [NetSuite SuiteSignOn Translation of OAuth Definitions](#).

For code samples that implement SuiteSignOn, see [Sample SuiteSignOn HTTP Calls](#).

NetSuite SuiteSignOn Translation of OAuth Definitions

When you are familiar with the OAuth specification (available at <http://oauth.net/documentation/spec>), refer to the following table to understand how the SuiteSignOn feature implements OAuth:

NetSuite Term	Definition	Analogous OAuth Term
Service Provider	NetSuite	Service Provider
Consumer	External application provider of the application to be accessed from NetSuite through SuiteSignOn, may also be known as partner .	Consumer
Consumer Key	Globally unique identifier of the consumer, generated by NetSuite.	Consumer Key
Shared Secret	Password used to establish ownership of the Consumer Key, entered by the Consumer when setting up the SuiteSignOn connection. The shared secret, also known as the signature , has a 1-1 relationship with the Consumer Key.	Consumer Secret
Token	Value used to gain access to protected resources on behalf of the user, generated by NetSuite, good for a single session.	
User	Individual who has logged into NetSuite and initiated activity between the consumer and NetSuite.	User

Mappings of dc and env URL Parameter Values

SuiteSignOn integration code dynamically creates values for the `dc` and `env` URL parameters.

The `dc` parameter ensures that integrations work for NetSuite accounts on all data centers.

The `env` parameter is used to identify whether the request is coming from a NetSuite production, release preview, or sandbox account. You might use this data to deny certain sessions (for example, requests coming from a sandbox account). Or you may want to map certain NetSuite requests to specific external applications. For example, you may want to map a request originating from the NetSuite sandbox account to a sandbox version of the external application.

The `dc` and `env` parameters apply the following:

- HTTP verify calls from external applications to NetSuite — see [External Application HTTP Verify Call](#)
- HTTP verify call responses from NetSuite to external applications — see [NetSuite HTTP Verify Call Response](#)

Both the `dc` and the `env` variables are required in order for your external application to map to a NetSuite domain.

For more information on `dc` and `env` parameter values, and URL examples, see the help topic [Understanding NetSuite URLs and Data Centers](#).

Sample SuiteSignOn HTTP Calls

As described in [SuiteSignOn Sequence Diagram and Connection Details](#), the SuiteSignOn handshake process between NetSuite and the external application includes the following calls in HTTP headers:

1. [NetSuite HTTP Outbound Call](#) sends the token and any context information to the external application.
2. [External Application HTTP Verify Call](#) returns the token and sends other required parameters to NetSuite. (This is the most important call for you to understand, as you will need to include it in your application's HTTP header.)
3. [NetSuite HTTP Verify Call Response](#) sends user identity information in XML format to the external application.

NetSuite HTTP Outbound Call

When a user accesses a SuiteSignOn connection point, NetSuite issues an outbound call to start the handshake. The following is an example of this call:

```
GET /SSO/demoApp.php?oauth_token=05016d16126a7a6c554656421e242310060807051b17ee54e6d26986d8aa&dc=001&env=PRODUCTION HTTP/1.1
Host: externalsystem.com
User-Agent: Mozilla/5.0 (Windows NT 6.1; WOW64; rv:19.0) Gecko/20100101 Firefox/19.0
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate
Connection: keep-alive
```

Note the following:

- This call uses the GET method to send a generated token to the external application.
- The external application, not the local host, is the host.
- This call includes a `dc` and an `env` parameter. These values can be mapped to the correct domain for NetSuite access. SuiteSignOn integration code should dynamically populate domains for NetSuite access based on the value of the `dc` and `env` parameters, to ensure that integrations work for NetSuite accounts on all data centers, and all types of accounts (for example, production accounts as opposed to sandbox accounts). See the help topic [Understanding NetSuite URLs and Data Centers](#) for more information on `dc` and `env` parameters, and URL examples.
- This call also may include context information, if integration variables have been defined for the connection point on the NetSuite SuiteSignOn page. `customer_id=970` is an example of an integration variable. It could be included as a URL parameter, as follows:

```
GET /SSO/demoApp.php?oauth_token=05016d16126a7a6c554656421e242310060807051b17ee54e6d26986d8aa&customer_id=970&dc=001&env=PRODUCTION HTTP/1.1
```

- URL parameters are separated by the ampersand (&). You need to make sure that your code properly parses these parameters. Your code should not rely on the number or order of URL parameters, as these are subject to change.

External Application HTTP Verify Call

Upon receipt of the NetSuite HTTP outbound call, the external application needs to issue an HTTP verify call. The following is an example of this call:

Note: HMAC-SHA256 is recommended, as it is the most secure signature option. You can also use HMAC-SHA1. PLAINTEXT is also supported.

```
GET /app/common/integration/ssoapplistener.nl HTTP/1.0
Host: system.netsuite.com
Authorization: OAuth oauth_consumer_key="60tBtQV4nmEQKpw", oauth_token="05016d16126a7a6c554656
421e242310060807051b17ee54e6d26986d8aa", oauth_nonce="kPeHzQpN6bZxSwu5w2nm", oauth_timestamp="1
490706743", oauth_signature_method="HMAC-SHA256", oauth_version="1.0", oauth_signature="vh3C69a
f9EwXKGbmlDqeA4xiYbtaM1Mq9WH60it4e5Q%3D"
```

Note the following:

- This call should use the GET method.
- This call should point to the NetSuite `ssoapplistener.nl` URL.
- The host domain should be dynamically populated with the system domain that maps to the `dc` and `env` parameter values. See the help topic [Understanding NetSuite URLs and Data Centers](#) for more information on `dc` and `env` parameters, and URL examples.
- This call should include the following parameters in the Authorization header. This entire header, including all of the parameters, should be in a single line.
 - **oauth_token** - The token generated and sent by NetSuite.
 - **oauth_consumer_key** - A globally unique identifier for the application provider, generated by NetSuite when the integration is set up on the SuiteSignOn page.
 - **oauth_signature_method** - HMAC-SHA256 and HMAC-SHA1 are supported signature methods for `ssoapplistener` and web service calls. HMAC-SHA256 is recommended. PLAINTEXT is also supported.
 - **oauth_signature** - The **shared secret**, entered when the integration is set up on the SuiteSignOn page. The shared secret is the password used to establish ownership of the consumer key, and has a 1-1 relationship with the consumer key. It should be percent-encoded, meaning it should be followed by the ampersand.
 - **oauth_timestamp** - The number of seconds since January 1, 1970 00:00:00 GMT. The timestamp value must be a positive integer and must be equal to or greater than the timestamp used in previous verify calls.
 - **oauth_nonce** - A random number that is unique across verify calls with the same timestamp value.

NetSuite HTTP Verify Call Response

Upon receipt of the verify call from the external application, NetSuite sends a response. The following is an example of this response:

```
HTTP/1.1 200 OK
Date: Tue, 16 Apr 2016 13:30:41 GMT
Server: Apache/2.2.17
Set-Cookie: lastUser=1326288_79_3; expires=Tuesday, 23-Apr-2016 13:30:42 GMT; path=/
Set-Cookie: NS_VER=2015.2.0; domain=system.netsuite.com; path=/"
```

```

X-Powered-By: Servlet/2.5 JSP/2.1
P3P: CP="CAO PSAa OUR BUS PUR"
Vary: User-Agent
Connection: close
Content-Type: text/html; charset=utf-8

<?xml version="1.0" encoding="UTF-8"?>
<outboundSso>
  <entityInfo>
    <ENTITYLASTNAME>Smith</ENTITYLASTNAME>
    <ENTITYINTERNALID>79</ENTITYINTERNALID>
    <ENTITYACCOUNT>1326288</ENTITYACCOUNT>
    <ENTITYFIRSTNAME>John</ENTITYFIRSTNAME>
    <ENTITYEMAIL>jsmith@netsuite.com</ENTITYEMAIL>
  </entityInfo>
</outboundSso>

```

Note the following:

- The domain set in the cookie is the same as the Host value in the external application HTTP verify call.
- The XML element formatting of fields is as name/value pairs, with element names formatted as follows: <ENTITYFIELDID> for standard fields, and <FIELDID> for custom fields.

Inbound Single Sign-on

The inbound single sign-on feature allows users to go directly from an external user-authenticating application to NetSuite, without having to log in separately to NetSuite. This feature allows a one-way trust relationship to be established between the external application and NetSuite, so that after users present login credentials to the external application, they can gain access to NetSuite as well.

Inbound single sign-on access generally has two types of users:

- Customers, who want to integrate their own NetSuite data with an external application's data.
- Application providers, who want to integrate customer data stored in their data center with their customers' NetSuite data.

With inbound single sign-on, authentication information from the external application is passed to NetSuite through an encrypted token, and a dynamically constructed URL redirects users from the external site to a NetSuite landing page. A mapping between each user's external credentials and their NetSuite credentials is created, either through a web services operation or through the user's login to NetSuite on their first single sign-on access.

To implement inbound single sign-on from your site to NetSuite, you must set up a trust relationship, with OpenSSL encryption keys, between your application and NetSuite. These keys are used to produce and interpret encrypted tokens. You also must write application code that dynamically constructs the redirect URL for each inbound single sign-on user. HTTP POST requests are not supported. NetSuite provides a downloadable kit with tools you can use for these tasks.

To get started with inbound single sign-on:

1. Review this guide, including the following, to ensure that the NetSuite inbound single sign-on feature will meet your needs.
 - For an overview of how inbound single sign-on works, see [Understanding Inbound Single Sign-on](#).
 - For instructions for setting up and implementing an inbound single sign-on integration, see [Setting Up Inbound Single Sign-on](#). This section includes the following information:
 - [Initial Setup for the Inbound Single Sign-on Feature](#)
 - [Implementing Inbound Single Sign-on in an External Application](#)
 - [Generating Keys Using OpenSSL](#)
 - [Creating the Initial Mapping of the Administrator Role for Inbound Single Sign-on](#)
 - [Creating Single Sign-on Code Using SSOUrl](#)
 - For instructions for setting up inbound single sign-on mappings, see [Mapping Users and Roles for Inbound Single Sign-on Access to NetSuite](#)
 - For technical background, see [Technical Summary of Inbound Single Sign-on](#).
2. Be aware of the following:
 - Inbound single sign-on access is supported for the NetSuite application, including the Customer Center, and for NetSuite web stores.
 - Inbound single sign-on access to the Customer Center is supported for NetSuite users classified as customers and for customer contacts.
 - Inbound single sign-on access to NetSuite respects IP address restriction rules. For information about this feature, see [Enabling and Creating IP Address Rules](#).
 - Inbound single sign-on access to web store is supported for custom checkout domains, multi-site implementations, and sites customized with SSP applications. Access is also supported

for Reference Cart & One Page Checkout, the NetSuite reference implementation of the web store checkout process. See the help topic [Inbound Single Sign-on Access to Web Store](#).

3. After you have confirmed that you want to implement inbound single sign-on, contact your account manager to purchase the feature.

Alternate Inbound Single Sign-on Mechanisms

NetSuite supports two other features that do not use this NetSuite version of token-based inbound single sign-on authentication:

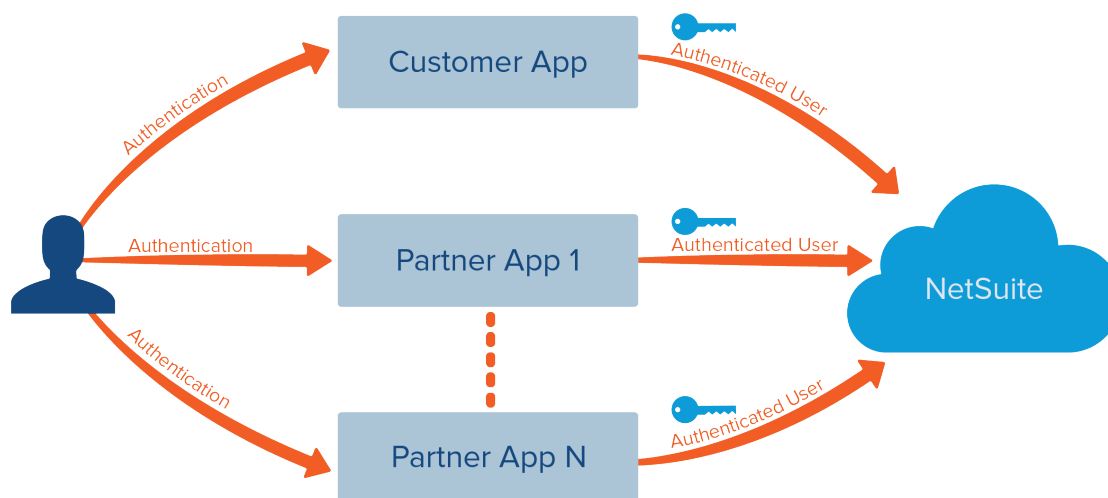
- The SAML Single Sign-on feature supports inbound single sign-on access to NetSuite using authentication from a SAML v2.0-compliant third-party identity provider. See [SAML Single Sign-on](#).

Note: It is not necessary to purchase the NetSuite Inbound Single Sign-on feature if you want to implement SAML Single Sign-on in NetSuite.

- The OpenID Single Sign-on feature supports inbound single sign-on access from Google Apps to NetSuite, relying on Google Accounts as the trusted system of authentication. See [OpenID Single Sign-on](#).

Inbound Single Sign-on Overview

The NetSuite inbound single sign-on feature enables users to move directly from an external user-authenticating web application to NetSuite without additional authentication. This feature provides token-based integration.

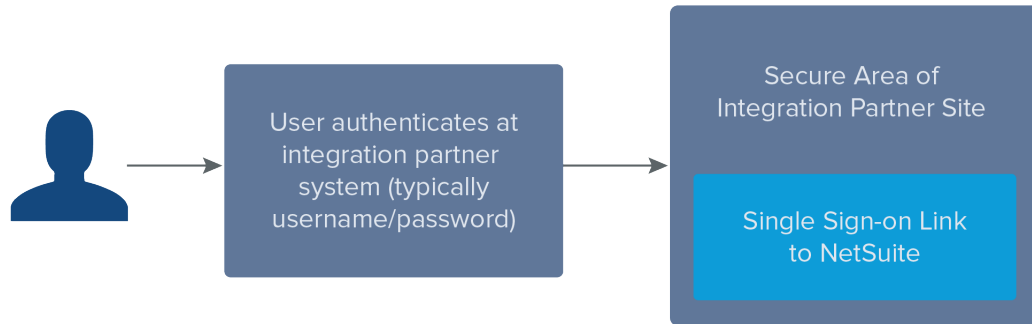


The external application uses an encrypted token to pass the user's identity to NetSuite, NetSuite verifies the token, then logs in the user. This single sign-on mechanism can be implemented through a link in the external application user interface, or through web services calls that use the [ssoLogin](#) operation.

Understanding Inbound Single Sign-on

The following steps outline how inbound single sign-on to NetSuite works.

1. In most cases, a user initiates inbound single sign-on access to NetSuite by clicking a link in an authenticated area of an external site. This site can be used in conjunction with either the NetSuite application user interface or a NetSuite web store.



Alternatively, the web services [ssoLogin](#) operation can be used to initiate inbound single sign-on access programmatically.

2. When a user initiates inbound single sign-on access, the external application produces a token that includes the following information:

- the user's external application company ID
This value is a string used by the external application to determine the company with which a user is associated, for example `ABCAutoParts`. It cannot contain spaces.
- the user's external application user ID
This value is a string used by the external application as a user identifier, for example `John.Smith`. It cannot contain spaces.
- the current timestamp
The timestamp string is a decimal representation of the number of milliseconds since January 1, 1970, 00:00:00 GMT.

For more information about the token, see the following topics:

- [Tables of Single Sign-on Redirect URL Parameters](#)
- [Elements of the Authentication Token String](#)
- [Example Inbound Single Sign-on Token](#)

3. The external application encrypts the information included in the token.
 - To encrypt the token, the external application must have access to a private key generated using OpenSSL. To interpret the encrypted token, NetSuite must have access to a public key extracted from this private key.
 - The inbound single sign-on kit includes Java classes you can use to produce the private and public keys. For instructions, see [Generating Keys Using OpenSSL](#).
4. After encryption of the token, the external application causes the user's browser to perform a redirect to NetSuite. HTTP POST requests are not supported.
 - The redirect is either to the NetSuite application or to the web store, based on the target set in the code (either `app` or `site`).
 - The redirect uses a URL constructed specifically for this inbound single sign-on access. This URL includes required parameters such as:
 - a hex-encoded, encrypted string representing the token
 - the unique partner ID assigned by NetSuite Customer Support
 - For NetSuite application access only, the remote company ID, which is the company ID used in the token

- For web store access only, the domain, NetSuite company ID, and site ID



Note: For more information on required and optional parameters, see [Tables of Single Sign-on Redirect URL Parameters](#).

- This URL is valid for up to 15 minutes after the timestamp included in the encrypted token.
 - The inbound single sign-on kit includes a Java class you can use to create code that dynamically constructs this URL. For instructions, see [Creating Single Sign-on Code Using SSOUrl](#).
5. NetSuite receives the token. Based on a unique partner ID assigned by NetSuite when inbound single sign-on is set up, NetSuite determines the public key that should be used to decrypt the token. After the token is decrypted, if the timestamp is valid, the request is honored.
 6. NetSuite checks for a user mapping between the external application and NetSuite.
 - If there is an existing mapping, the user is logged in as defined by the mapping.
 - If a mapping does not exist, the user is prompted to provide NetSuite credentials to create the mapping.



Important: A user with an Administrator role must create the initial mapping from the external application to NetSuite. For more information, see [Creating the Initial Mapping of the Administrator Role for Inbound Single Sign-on](#).

After the initial mapping to the Administrator role is completed:

- For web store access, the administrator is required to use the web services [mapSso](#) operation to create the account mapping for multiple users so that it is available before users initiate single sign-on access.
 - For NetSuite access, the administrator can use the web services [mapSso](#) operation to create the account mapping for multiple users, or can instruct users to create their own mappings. See [Mapping Users and Roles for Inbound Single Sign-on Access to NetSuite](#).
7. After the user's identity has been verified, a NetSuite landing page displays.
 - The default landing page for NetSuite application access is the user's home page.
 - The default landing page for web store access is the site home page.
 - A non-default landing page can be defined by adding a `landingurl` parameter to the redirect URL.
 - If no mapped NetSuite identity is found, the NetSuite inbound single sign-on login page displays.
 8. A NetSuite session initiated through inbound single sign-on is subject to standard NetSuite session timeout rules.
 - By default, the user is redirected to an inbound single sign-in login page on session timeout or error.
 - This NetSuite login page can be hidden by setting the redirect URL's `hideloginpage` parameter to true, so that the user is returned to a different page, such as an external application page. In this case, a `returnurl` parameter also must be added, to specify this alternate page.
 9. The user can log out from an inbound single sign-on session in the same manner as any other NetSuite session.
 - By default, the user is redirected to the inbound single sign-in login page on logout.

- If the redirect URL includes the `returnurl` parameter, the user is redirected to the page specified by this parameter instead. It is not necessary to set the `hideloginpage` parameter to `T` (true) to vary the page on logout.

Setting Up Inbound Single Sign-on

See the following two procedures for important information:

1. Initial Setup for the Inbound Single Sign-on Feature

This required procedure outlines requirements for setting up the inbound single sign-on feature in your account. It is guided by NetSuite Customer Support, and occurs after you have contacted your NetSuite account representative and purchased the Inbound Single Sign-on feature.

2. Implementing Inbound Single Sign-on in an External Application

This procedure outlines options for inbound single sign-on integration from an external application to NetSuite.

Note: It is not necessary to purchase the NetSuite Inbound Single Sign-on feature if you want to implement SAML Single Sign-on in NetSuite. For more information, see [SAML Single Sign-on](#). See also [Alternate Inbound Single Sign-on Mechanisms](#).

Initial Setup for the Inbound Single Sign-on Feature

Warning: You must contact your NetSuite account representative and purchase the inbound single sign-on feature to properly initiate the setup process. Do not attempt to complete these steps on your own. Wait until you are contacted by NetSuite Customer Support to begin the initial setup of this feature.

To complete the initial setup of the inbound single sign-on feature in your account:

1. Contact your account representative to purchase the inbound single sign-on feature.
 - a. NetSuite Customer Support will open a new support case, and contact you for specific information.
 - b. NetSuite Customer Support will ask you to generate a public and private key pair using OpenSSL. See [Generating Keys Using OpenSSL](#).
 - c. NetSuite Customer Support will ask you to provide the generated public key through the support case.
 - d. NetSuite Customer Support will associate the public key with an Inbound Single Sign-on Partner ID, and will provide this unique Partner ID to you.
2. After NetSuite Customer Support guides you through the initial setup, and provides you with your unique Partner ID, you will be ready to implement inbound single sign-on in your application. See [Implementing Inbound Single Sign-on in an External Application](#).

Implementing Inbound Single Sign-on in an External Application

Before you attempt to implement any of the following options in your external application, you must have already contacted your NetSuite account administrator and purchased the Inbound Single Sign-on

feature. Also, NetSuite Customer Support must have already guided you through the steps outlined in [Initial Setup for the Inbound Single Sign-on Feature](#).

To implement inbound single sign-on in an external application:

You can choose any of the following options to implement inbound single sign-on from an external application to NetSuite:

1. Download the kit for implementing inbound single sign-on:

<https://system.netsuite.com/download/NLSingleSignOn.zip>.



Note: A checksum file is also available: <https://system.netsuite.com/download/NLSingleSignOn.sha512>.

2. Add the `ssov3.jar` file from this kit to your Java classpath.

You need the contents of this .jar file to facilitate compilation of your single sign-on integration code and to generate keys for token encryption.



Note: Java developers can add the `ssov3.jar` to your classpath, along with the Java run-time environment classes. Source code is also provided for developers in non-Java environments as a template for implementation.

3. Write application code that dynamically constructs redirect URLs to be used when users initiate inbound single sign-on access. HTTP POST requests are not supported. See [Creating Single Sign-on Code Using SSOUrl](#).
4. Write web services code for the single sign-on integration as needed.
You can programmatically initiate access with `ssoLogin`, and/or programmatically map users' external credentials to NetSuite credentials. See [SuiteTalk \(Web Services\) Single Sign-on Operations](#).
5. Provide error handling for status codes returned from NetSuite inbound single sign-on sessions. See [Error Handling for Inbound Single Sign-on](#).
6. To prevent single sign-on users from directly logging in to NetSuite, create a custom role that is designated as **Single Sign-on Only**, and assign this role to single sign-on users. See [Setting Up a Single Sign-on Only Role](#).

Generating Keys Using OpenSSL

As described in the section [Initial Setup for the Inbound Single Sign-on Feature](#), NetSuite Customer Support will ask you to generate a public and private key pair using OpenSSL. The public key is provided to NetSuite, for use in creating your unique Partner ID. You will use the private key in your implementation to encrypt authentication tokens.

To generate keys for inbound single sign-on:

1. Either append the `openssl` subdirectory provided in the inbound single sign-on kit to your PATH, or download source code from <http://www.openssl.org/source>.

The binaries included in the `openssl` directory are derived from `openssl1.0.9.6.tar.gz`. If you are creating your own binaries from a downloaded source package, follow directions in the **INSTALL** file appropriate to your operating system.

2. After `openssl` is installed and in your PATH, type `openssl` to get the following prompt:

```
OpenSSL>
```

3. At the prompt, use the following command to generate a private key:

```
OpenSSL> genrsa -out <privKey.pem> -rand <f1><s><f2><s><f3><s><f4><s><f5> 2048
```

- <privKey.pem> is the desired name of the output file.
 - <f1> through <f5> are names of files used as random seeds.
 - <s> is a separator:
 - ; for Windows
 - , for OpenVMS
 - : for all other operating systems
 - This process generates a private key with a modulus length of 2048 bits. The output file format produced (PEM) is not appropriate for use by NetSuite tools, however, and must be properly formatted, as described in the following step.
4. Convert the private key to DER using the `Pem2Der` class provided by NetSuite, by typing the following Java command:

```
java com.netledger.forpartners.encryption.Pem2Der <privKey.pem>
<privKey.der>
```

5. Extract the public key from the private key using the `Priv2Pub` class provided by NetSuite, by typing the following command:

```
java com.netledger.forpartners.encryption.Priv2Pub
<privKey.der> <pubKey.der>
```

- <privKey.der> and <pubKey.der> of this last command are your public and private keys.

Note: As described in the section [Initial Setup for the Inbound Single Sign-on Feature](#), you will provide the public key to NetSuite Customer Support through the support case. Maintain the private key for use by the NetSuite `SSOUrl` class. See [Creating Single Sign-on Code Using SSOUrl](#).

Creating the Initial Mapping of the Administrator Role for Inbound Single Sign-on

The initial single sign-on account mapping must be a mapping to an Administrator role in NetSuite. This Administrator role mapping serves as authorization that NetSuite trusts the external authentication system. This requirement gives NetSuite administrators control over when single sign-on functionality is available to their users.

Note: The following procedure assumes that you have completed all tasks as described in the [Initial Setup for the Inbound Single Sign-on Feature](#) and [Generating Keys Using OpenSSL](#) topics. (You have added the `ssov3.jar` file from the inbound single sign-on kit to your Java classpath, NetSuite Customer Support has assisted with the generation of the public and private keys, and NetSuite Customer Support has provided you with the Partner ID for this account.)

A user with an Administrator role in this account must create the initial mapping between NetSuite and the external authentication system. To generate the token necessary to complete the initial mapping, you can use the web services `mapSso` operation, or the external third-party application if it is available. You can also use the tool included in the Inbound Single Sign-on kit to generate the URL, which includes the token necessary for this procedure.

To create a token with the ssov3.jar file:

1. Call the `ssov3.jar` file.
2. Specify the appropriate parameters:
 - a. `rc` = the target account number.
 - b. `p` = the Partner ID assigned by NetSuite Customer Support.
 - c. `ru` = the Administrator role to be used for the initial mapping.
 - d. `t` = the type of URL you want to generate, either `app` (for inbound access to the NetSuite UI) or `site` (for inbound access to your website).

See [Tables of Single Sign-on Redirect URL Parameters](#) for more information.

3. The tool generates a URL with a token in the form: `<domain>/app/login/secure/sso.nl<partnerID><TokenBigLongString>`

Note: The token is valid for 15 minutes. You must complete the following steps before the token expires, or generate a new token.

4. Copy and paste the URL into your browser's address bar and click **Enter**. The NetSuite Partner Login page displays.
5. On the NetSuite Partner Login page, enter your NetSuite email address and password.
6. Click **Log in**. The NetSuite Choose a role to create the mapping page displays.
7. Click your Administrator role for this account.

Note: If you have Administrator roles in more than one account, ensure you are selecting the correct Administrator role for this specific account.

The initial mapping of the Administrator role is now complete.

After the initial mapping is completed, other users and roles can now be mapped to the external application. The web services `mapSso` operation can be used to create the account mappings for multiple users so that they are available before users initiate single sign-on access. (This method of mapping is required for web store access.)

Creating Single Sign-on Code Using SSUrl

The quickest way to create inbound single sign-on code is to use the `SSUrl` Java class provided in the downloadable kit. This class is available to you after you have downloaded the kit and added the `ssov3.jar` file to your Java classpath.

The `SSUrl.java` file provides a template for Java code, along with explanatory comments. You can use this file to guide your creation of single sign-on integration code in Java. A command-line utility that you can run from a shell is also provided as an alternative.

Note: If you are NOT using the NetSuite `SSUrl` class to generate redirect URLs, then you will need to construct them using your own methods from the base elements described in `SSUrl.java`.

If you are using the `SSUrl` class to implement inbound single sign-on integration, your application code needs to do the following:

- Initialize the `SSUrl` class with the file name of the private key used to encrypt the authentication token.
- Set the target of the inbound single sign-on access to either the NetSuite application (`app`) or the web store (`site`), so that the base URL for the integration is correctly generated:

- for the NetSuite application:

```
https://system.netsuite.com/app/login/secure/sso.nl
```

HTTP POST requests are not supported.

- for the web store:

```
https://checkout.mycompanystore.com/app/site/backend/sitesso.nl
https://checkout.netsuite.com/app/site/backend/sitesso.nl
https://checkout.na1.netsuite.com/app/site/backend/sitesso.nl
https://checkout.na2.netsuite.com/app/site/backend/sitesso.nl
```

- For inbound single sign-on access to web store, set the domain for your web store.
 - If you have a custom checkout domain, set the domain appropriately.
 - If you are using a NetSuite-hosted checkout domain, set it to the appropriate checkout domain for your data center. The NetSuite company ID and site ID URL parameters are also required for web store access. For more information, see [Web Store Access Only Parameters](#).
- Provide the single sign-on link as a link to an internal page that uses the `SSOUrl.getURL(companyId, userId)` method to dynamically construct a redirect URL to a landing page in the NetSuite application or web store. This URL should include all required parameters and any desired optional parameters. HTTP POST requests are not supported.
- Redirect the browser to the constructed URL.

For more information, see the following:

- [Tables of Single Sign-on Redirect URL Parameters](#)
- [Example Single Sign-on Token and Redirect URLs](#)

Tables of Single Sign-on Redirect URL Parameters

The following tables describe the parameters used in single sign-on redirect URLs. HTTP POST requests are not supported.

Review all of the tables to ensure that you are including all of the parameters required for your purposes.

- [Parameters used for both Application and Web Store Access](#)
- [NetSuite Application Access Only Parameters](#)
- [Web Store Access Only Parameters](#)

In addition, see the following:

- For further insight into parameter usage, review the contents of the `SSOUrl.java` file.
- For example redirect URLs, see [Example Single Sign-on Token and Redirect URLs](#).

Parameters used for both Application and Web Store Access

The following table describes the parameters used for both the NetSuite application access (`app`) and web store access (`site`).

Name	Description	Required/Optional	Programmatic Parameter	Command-Line Parameter
authentication token	string representing the encrypted token	Required	a	

Name	Description	Required/Optional	Programmatic Parameter	Command-Line Parameter
	For more information, see Elements of the Authentication Token String . See also Example Inbound Single Sign-on Token .			
partner ID	unique ID assigned by NetSuite for use with inbound single sign-on; this ID is associated with the public key you provided to NetSuite	Required	pid	p
landingurl	page that first displays for inbound single sign-on access, other than default Defaults are: <ul style="list-style-type: none"> for NetSuite application access, the user's home page for web store access, the site home page 	Optional	landingurl	l
hideloginpage	Boolean indicating whether to hide the default inbound single sign-in login page from users and instead go to the page specified by the returnUrl parameter By default, set to false. NetSuite recommends that it be set to true for web store access.	Optional	hideloginpage	h
returnurl	page where single sign-on users are redirected on session logout, timeout, and errors Default is the inbound single sign-on login page.	Required if hideloginpage set to true	returnurl	r

Elements of the Authentication Token String

The format of the authentication token string prior to encryption is:

```
<companyID><space><userID><space><timestamp>
```

The `companyID` and `userID` elements represent the credentials used by the external application. (These credentials will be mapped to the NetSuite identity during inbound single sign-on.) Because spaces are used to delimit subtokens, `companyID` and `userID` elements cannot contain spaces.

See the following information about each element in the string:

- The `companyID` element is used by the external application to determine the company with which a user is associated, for example, `ABCAutoParts`. The `companyID` that you should use can vary. The goal is to ensure that the application token string is unique.
 - If you are a partner building an application for another company, the `companyID` should be a unique identifier of that company. You could use the company's name, or any identifier you use to identify that company.

- If you are building an integration for your own company, use your company name.
- In any case, you can always use the NetSuite account ID as the `companyID`. To locate the account ID, go to Setup > Company > Setup Tasks > Company Information. The account ID field is located near the bottom of the right column.
- The `userID` string used by the external application as a user identifier, for example, `John.Smith`. It cannot contain spaces.
- The `timestamp` string is a decimal representation of the number of milliseconds since January 1, 1970, 00:00:00 GMT. The token is valid for 15 minutes after the timestamp contained in it.

NetSuite Application Access Only Parameters

The following table describes the NetSuite application access only parameters.

Name	Description	Required/Optional	Programmatic Parameter	Command-Line Parameter
partner account / remote company ID	External application-assigned ID for your company. This value is identical to the <code>companyID</code> value used in the token. For more information about the <code>companyID</code> , see Elements of the Authentication Token String .	Required if target is app	pacct	rc
partner user ID / remote user ID	External application-assigned ID for your user. This value is identical to the <code>userID</code> value used in the token.	Required if target is app	puid	ru
application domain	Allows specification of domain and data center, for example, <code>system.na1.netsuite.com</code> . If specified, the application domain for the base URL will be overridden with provided value.	Optional		ad

Web Store Access Only Parameters

The following table describes the web store only access parameters used in single sign-on redirect URLs. These parameters are used when the target is `site`. HTTP POST requests are not supported.



Important: Always specify the NetSuite-hosted checkout domains (including the correct data center) in addition to the `c` parameter for faster routing if the performance of the login operation is a concern.

Name	Description	Required/Optional	Programmatic Parameter	Command-Line Parameter
domain	Your custom checkout domain, for example: <code>checkout.mycompany.com</code> . If you use a NetSuite-hosted checkout domain, enter the domain. The domain must include the correct data center identifier.	Required for custom checkout domains. Recommended for NetSuite-hosted checkout domains.		d

Name	Description	Required/Optional	Programmatic Parameter	Command-Line Parameter
	See Checkout Domains for Data Centers .			
NetSuite company ID	NetSuite-assigned account ID for your company	Required for NetSuite-hosted checkout domains.	c	c
site ID	internal ID on a NetSuite website record; distinguishes among multiple sites in your web store integer displayed on the Web Site Preview page, as shown in Finding the Site ID Parameter A site ID of 1 is valid even with only a single site.	Required when the c parameter is used.	n	s

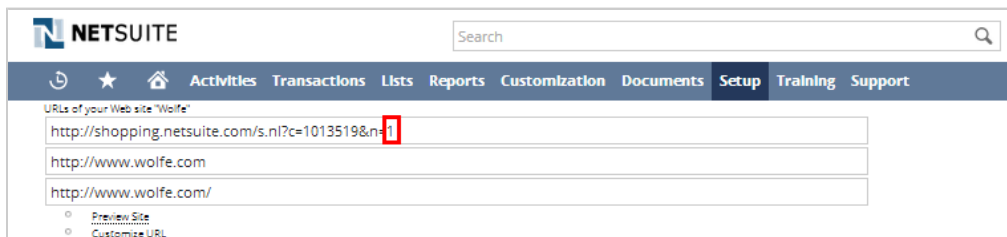
Checkout Domains for Data Centers

If you use a NetSuite-hosted checkout domain, for the domain parameter, use the correct data center identifier in your URL.

- NA West: `checkout.netsuite.com`
- NA East: `checkout.na1.netsuite.com`
- NA Northwest: `checkout.na2.netsuite.com`
- NA Central: `checkout.na3.netsuite.com`
- EU West: `checkout.eu1.netsuite.com`
- EU Central: `checkout.eu2.netsuite.com`

Finding the Site ID Parameter

You can find the value of the site ID parameter to set for a multi-site environment at Setup > Site Builder > Preview Web Site:



Note: A site ID of 1 is valid for a single site as well as for a multi-site environment.

Example Single Sign-on Token and Redirect URLs

Review the following examples to get a better understanding of inbound single sign-on tokens and redirect URLs:

- [Example Inbound Single Sign-on Token](#)

- Example Redirect URL for the NetSuite Application
- Example Redirect URL for the Web Store
- Example Redirect URL for Intermediate Third-party Login to Web Store

For details about the parameters used in redirect URLs, see [Tables of Single Sign-on Redirect URL Parameters](#).

Example Inbound Single Sign-on Token

The following example illustrates the three stages of generating an inbound single sign-on token. The token is created with:

- the external application-assigned remote company ID, and
- the remote user ID for the user, and
- the current timestamp.

The token is then encrypted using a private key, and then hex-encoded so it can be passed as a redirect URL parameter.

Note: The hex-encoded, encrypted token string that is used as the URL parameter.

Stages of Token Generation

1. Plain Text String:

ABCAutoParts John.Smith 1225479286770

2. Encrypted String:

```
W4$ŮŮ7æ½anzóŭ-°Pm 3vx'kaD†'LĀŌI 'aZusYāqPocC~Ō'ĬC)Ů¼X
pRó'vôĪĀD¼Ů~¼āĀi)Ů...y×1Ōj Ō'Ō*Ŋgfaq'Ů±jā'XôCi<(ĀBŭŝ7
```

3. Hex-Encoded, Encrypted String:

```
57E1A7DA1CD637E6BD1F6D7AF3F9EE96B0DE6D1E0D337678606BE4448760CEC5D249031CA0B4618EB50C731703DDE27
150F663C7AC11D3B4CEB84329DB2EBE58FE1D16520FF3271476F069C31DD0BCDA0A0AFB4BEF1C3EC0F7DD98579D7314F
6A0AD5B3D22AD1A2E766E471B0FA22B1BFED4Br58Frr315EC3C7BC1C65CFA9A37
```

Example Redirect URL for the NetSuite Application

The following is an example redirect URL for inbound single sign-on access to the NetSuite application:

```
https://system.netsuite.com/app/login/secure/sso.nl?pid=198765&pacct=ABCAutoParts&puid=John.Smi
th&a=57E1A7DA1CD637E6BD1F6D7AF3F9EE96B0DE6D1E0D337678606BE4448760CEC5D249031CA0B4618EB50C731703
DDE27150F663C7AC11D3B4CEB84329DB2EBE58FE1D16520FF3271476F069C31DD0BCDA0A0AFB4BEF1C3EC0F7DD98579D
7314F6A0AD5B3D22AD1A2E766E471B0FA22B1BFED4Br58Frr315EC3C7BC1C65CFA9A37
```

The base URL in the redirect URL is determined by the target set in integration code; in this case, the target is set to `app`.

Values for the URL parameters in this example also are set in integration code: the partner ID (`pid`), the remote company id (`pacct`), the remote user ID (`puid`), and the hex-encoded encrypted token string (`a`).



Important: The parameters listed above are valid parameters for this use case. For more information, see the [Tables of Single Sign-on Redirect URL Parameters](#).

Do not use the `ck` and `cktime` parameters described in the [Example Redirect URL for Intermediate Third-party Login to Web Store](#).

Example Redirect URL for the Web Store

The following is an example redirect URL for inbound single sign-on access to the web store:

```
https://checkout.netsuite.com/app/site/backend/sitesso.nl?a=57E1A7DA1CD637E6BD1F6D7AF3F9EE96B0D
E6D1E0D337678606BE4448760CEC5D249031CA0B4618EB50C731703DDE27150F663C7AC11D3B4CEB84329DB2EBE58FE
1D16520FF3271476F069C31DD0BCDA0A0AFB4BEF1C3EC0F7DD98579D7314F6A0AD5B3D22AD1A2E766E471B0FA22B1BFE
DE4Br58Frr315EC3C7BC1C65CFA9A37&pid=198765&hideloginpage=T&returnurl=http://www.abcautoparts.co
m/&c=198765&n=1
```

The base URL in the redirect URL is determined by the target set in integration code; in this case, the target is set to `site`.

Values for the URL parameters in this example also are set in integration code: the hex-encoded encrypted token string (`a`), the partner ID (`pid`), the hide login page indicator (`hideloginpage`), the return URL (`returnurl`), the NetSuite-assigned company ID (`c`), and the site id (`n`).

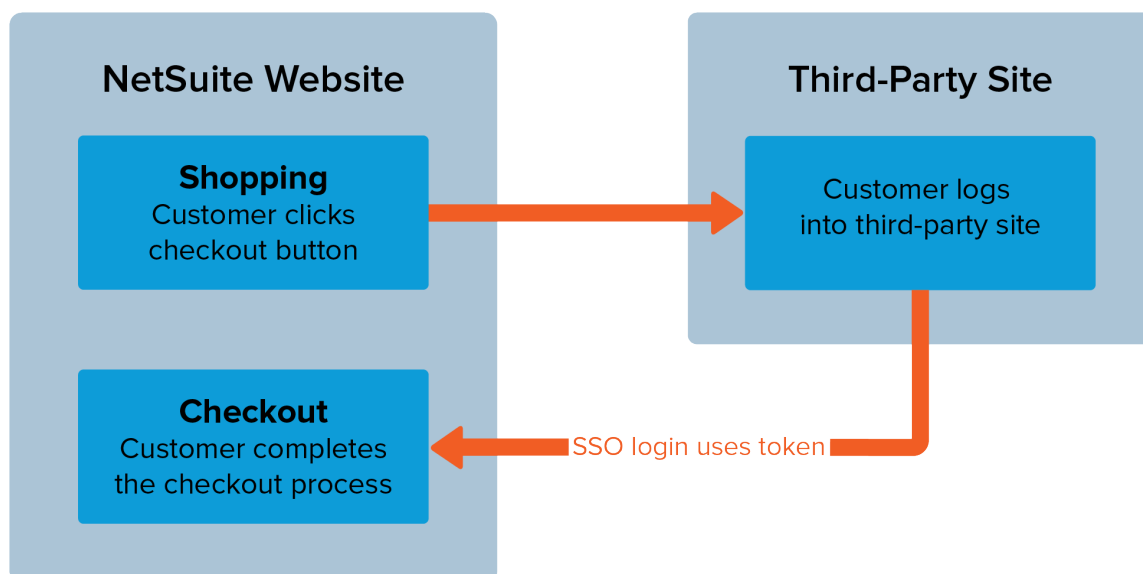


Important: The parameters listed above are valid parameters for this use case. For more information, see the [Tables of Single Sign-on Redirect URL Parameters](#).

Do not use the `ck` and `cktime` parameters described in the [Example Redirect URL for Intermediate Third-party Login to Web Store](#).

Example Redirect URL for Intermediate Third-party Login to Web Store

Inbound single sign-on supports the workflow where a customer visits a NetSuite shopping site, adds an item to the cart, clicks the Checkout button, and then is directed to a third-party site for login. As shown in the diagram below, after logging into the third-party site, the customer is directed to NetSuite checkout servers to complete a transaction.



To ensure that the shopping cart contents persist to the NetSuite checkout servers, parameters that allow the checkout servers to determine the original session must be included in the single sign-on call to NetSuite.



Important: The `ck` and `cktime` parameters described in the following procedure should only be used in situations when there is an intermediate third-party login required before proceeding to the Web Store.

To ensure synchronization between the NetSuite web store checkout server and the shopping server:

1. Include two additional parameters, `ck` and `cktime`, in the customized checkout link pointing to third-party servers for login. You can include these parameters by using tags in the customization text.

You might, for example, put the tags directly into the URL you are substituting for the checkout URL as:

```
&ck=< _NLSHOPPERID_>&cktime=< _NLCOOKIETIMESTAMP_>
```

2. Upon receiving these parameters on the third-party login resource, read them, and then save them for addition to the URL to which the customer is redirected for single sign-on after login at the third-party site.

Example:

```
https://checkout.netsuite.com/app/site/backend/sitesso.nl?landingurl=http%3A%2F%2Fshopping.f.ne
tsuite.com%2Fs.nl%3Fc%3D1035737&pid=1&c=1035737&a=792C4B61EF9BE695E9E9375FD78D24F25200EDEF01A4
16B03A2AAC41EE0E2C31F4503D33F0E7FED1C154BFD559B7AC9D8E1B9DE4B9882D4FF9488DB11867BCE03B1A91C9388
1B09F1FB99B0837BA0642CB58EA8B9839308503DF3ADDE3DD3F22ED37704D7C30171871C6439E0F69BCA49C6DAA2B5B
1D2651490B6FA4E3FA4BB&ck
=rBDDSZm-AdboaPPb&cktime=114233
```

The base URL in the redirect URL is determined by the target set in integration code; in this case, the target is set to `site`.

Values for the URL parameters in this example also are set in integration code: the page that first displays for inbound single sign-on access (`landingurl`), the partner ID (`pid`), the NetSuite-assigned company ID (`c`), the hex-encoded encrypted token string (`a`), the shopperid (`ck`), and a time stamp (`cktime`).



Important: The parameters listed above are valid parameters for this use case. For more information, see the [Tables of Single Sign-on Redirect URL Parameters](#). The `ck` and `cktime` parameters are additional parameters valid for this use case only.

SuiteTalk (Web Services) Single Sign-on Operations

The web services `mapSso` operation provides the ability to automate the mapping between users' external application credentials and NetSuite credentials.

- This operation provides inbound single sign-on access to NetSuite without users having to log in to NetSuite the first time this access occurs. Instead of the mapping between their external application credentials and NetSuite credentials being created at the time of this login, the mapping is created through web services.

- Use of this operation is required for inbound single sign-on access to the web store.
- This operation is applicable to accounts that have inbound single sign-on set up, and that have access to the associated external application credentials and encryption keys needed to generate the token.
- For more information, see the web services [mapSso](#) topic, which includes code samples.

Single sign-on mappings are not copied from a production account to a sandbox account when the sandbox is refreshed. These mappings must be recreated in the sandbox account for any users who require inbound single sign-on access to that account.

The web services `login` operation provides a mechanism for external applications to automate inbound single sign-on user login to NetSuite without the user's NetSuite credentials going through the external servers.

- This operation provides inbound single sign-on access to NetSuite without users having to click a link in the external application. The activities that occur when a user clicks this type of link instead occur behind the scenes.
- This operation is applicable to users who authenticate to NetSuite through the web services `login` operation; it is not applicable to users who authenticate to NetSuite by providing user credentials in the header of their SOAP requests.
- For more information, see the web services [ssoLogin](#) topic, which includes code samples.



Important: NetSuite hosts customer accounts in multiple data centers. For that reason, the correct URL for web services access varies depending on the data center hosting the account. Your integration must incorporate logic that dynamically determines the correct URL. With the 2012.2 and later endpoints, you should use the [getDataCenterUrls](#) operation to dynamically discover the correct URL. With older endpoints, you should use the REST roles service. For details, see the help topic [Using the REST roles Service to Get User Accounts, Roles, and Domains](#).

Error Handling for Inbound Single Sign-on

When an inbound single sign-on session is interrupted, NetSuite sends status codes to the external application. The external site can receive each of these status codes and display an appropriate error to the user.

- The following status codes may be returned if the `hideloginpage` parameter is set to T (true): (If `hideloginpage` is set to F (false), the user is redirected to the login page.)
 - `LOGIN_ERR_NO_MAPPING` - No SSO mapping of user authentication exists in NetSuite.
 - `LOGIN_ERR_UNKNOWN` - Unexpected error occurred.
 - `SESSION_TIMEOUT` - Session timed out in NetSuite.

Note that each inbound single sign-on token includes a timestamp, and single sign-on access is only valid for 15 minutes.
- The following status code may be returned independently of the `hideloginpage` parameter value:
 - `LOGOUT` - User chose to log out.

Setting Up a Single Sign-on Only Role

For security purposes, you can designate a NetSuite role as Single Sign-on Only. When a user logs in with a role that has been designated as Single Sign-on Only, validation is performed to ensure that the

user is logging in through an inbound single sign-on mechanism. This mechanism can be either the NetSuite certificate-based inbound single sign-on feature or OpenID single sign-on feature.

The Single Sign-on Only role supports strict control of credentials from the external application. This type of role increases the security of an integrated application by prohibiting a web services or UI user from accessing the system with permissions and privileges that are specifically created for inbound single sign-on access only.



Important: You cannot use NetSuite for Outlook with a Single Sign-on Only role. Users who are not sure whether their role is compatible with NetSuite for Outlook should contact their account administrator.

To designate a role as Single Sign-on Only:

1. Go to Setup > Users/Roles > Manage Roles.
2. On the Manage Roles list page, select **Edit** or **Customize** next to the role you want to set as **Single Sign-on Only**.
3. Check the **Single Sign-on Only Role** box.
4. Click **Save**.

Next, assign this role to single sign-on users as needed.

For details about setting up roles in NetSuite, see the help topic [Customizing or Creating NetSuite Roles](#).

Mapping Users and Roles for Inbound Single Sign-on Access to NetSuite

NetSuite verifies a user's identity by comparing the remote system credentials passed in the token (company ID and user ID) to their NetSuite credentials (email, password, account, and role used to log in to NetSuite). To allow for this comparison and verification, the remote system credentials must be associated with, or mapped to, the NetSuite credentials. This mapping stores a permanent association between the user's external application identity and their NetSuite identity.

The initial single sign-on account mapping must be to an Administrator role in NetSuite. This administrator mapping serves as authorization that NetSuite trusts the external authentication system. This requirement gives NetSuite administrators control over when single sign-on functionality is available to their users. See [Creating the Initial Mapping of the Administrator Role for Inbound Single Sign-on](#) for more information.

After the initial mapping to the Administrator role is completed:

- For web store access, the administrator is required to use the web services [mapSso](#) operation to create the account mappings for multiple users so that they are available before users initiate single sign-on access.
- For NetSuite access, the administrator can use the web services [mapSso](#) operation to create the account mappings for multiple users, or can instruct users to create their own mappings.

If the administrator does not create mappings for users' external credentials and NetSuite credentials, users are required to create these mappings when they first log in with a role that requires single sign-on access. (This method of mapping is not supported for web store access.) See [Creating Your Mapping for Inbound Single Sign-on to the NetSuite UI](#)

Be aware of the following:

- If a NetSuite role used for inbound single sign-on access is deleted, the single sign-on mapping for any user with that role is automatically remapped to another role.
- If a user has a single sign-on mapping set up with a particular role and that role is removed from the user, the mapping is deleted. You can set up a new mapping for that user with a different role.
- Single sign-on mappings are not copied from a production account to a sandbox account when the sandbox is refreshed, or from one sandbox account to another. These mappings must be recreated in each sandbox account for any users who require inbound single sign-on access to that account.

Note: If a user requires single sign-on access for multiple accounts, you must use a different partner account/ remote company ID for each single sign-on mapping for that user. For more information, see [NetSuite Application Access Only Parameters](#).

Creating Your Mapping for Inbound Single Sign-on to the NetSuite UI

If your NetSuite administrator did not already create the mapping for you, the first time you log in from an external application to the NetSuite UI in your inbound single sign-on role, you must create a mapping.

Note: This procedure is only for mapping access to the NetSuite application (the UI). The mapping for access to a web store must be completed by an account administrator using the `mapSso` operation.

To create your mapping for inbound single sign-on to the NetSuite UI:

1. Log in to the external application to be used for inbound single-sign-on access to NetSuite.
2. Click the link to go to the NetSuite UI. The NetSuite Partner Login page displays.
3. On the NetSuite Partner Login page, enter your NetSuite email address and password.
4. Click **Log in**. The NetSuite Choose a role to create the mapping page displays.
5. Click the name of the role you will use for inbound single sign-on access to this account.

Note: If you have similar roles in more than one account, ensure that you are selecting the correct role for this specific account.

The mapping is now complete. You will automatically be logged in when accessing NetSuite by inbound single sign-on from your external application.

Technical Summary of Inbound Single Sign-on

In the following text, system A refers to an external application, and system B refers to NetSuite.

Note: HTTP POST requests are not supported.

This example discusses two systems, systems A and B, and a user that has identifiers ID_A and ID_B in the respective systems. In the absence of single sign-on, A and B would each require ID and PASSWORD presentation for user access. In order for system A to validate the user and then use single sign-on to redirect the user to system B, without requiring further user authentication, the following steps occur:

1. System A validates user ID_A as usual, requesting PASSWORD_A.

2. User works in system A and eventually clicks a link to system B.
3. System A creates a string $T = ID_A + " " + TimeStampString$.
 - ID_A is $\langle companyID \rangle \langle space \rangle \langle userID \rangle$, so the entire token prior to encrypting and hex-encoding is $\langle companyID \rangle \langle space \rangle \langle userID \rangle \langle space \rangle \langle timestamp \rangle$. The $\langle companyID \rangle$ in system A maps to a similar ID in system B.
 - Because a $\langle space \rangle$ is used to delimit subtokens in the token, none of the subtokens may contain $\langle space \rangle$ characters.
 - The timestamp string is a decimal representation of the number of milliseconds since January 1, 1970, 00:00:00 GMT.
4. System A encrypts T using RSA encryption with a private key, KA_{p_r} , creating a token $\{T\}KA_{p_r}$.
5. System A hex-encodes the encrypted bits so they can be transported as a URL parameter, the result being $hex(\{T\}KA_{p_r})$.
6. System A directs the user's browser to a landing link on system B, including $hex(\{T\}KA_{p_r})$ as a URL parameter.
7. System B hex decodes $hex(\{T\}KA_{p_r})$, yielding $\{T\}KA_{p_r}$.
8. System B uses the public key, KA_{p_u} , corresponding to KA_{p_r} to retrieve T from $\{T\}KA_{p_r}$.
9. System B checks that T was recently generated by observing $TimeStampString$. This check reduces the risk of the token being used outside the context of single sign-on between A and B.
10. System B looks up ID_B in a table that maps $\{A, ID_A\} ID_B$.
11. System B trusts system A's authentication procedure, and therefore logs user ID_B into system B transparently.

SAML Single Sign-on

SAML (Security Assertion Markup Language) is an XML-based standard that supports communication of user data among different enterprise applications, called service providers (SPs). An identity provider (IdP) makes security assertions consumed by other service providers. A single IdP can perform user authentication for many SPs. A particular SP and an IdP can establish a circle of trust by providing each other with metadata in an XML format defined by SAML specifications, so that the SP accepts users authenticated by the IdP.

The NetSuite SAML Single Sign-on feature is based on the SAML v2.0 specifications. For information about these specifications, click [here](#). Any SAML 2.0-compliant application can serve as the IdP for SAML access to NetSuite.

The SAML Single Sign-on (SSO) feature supports inbound single sign-on access to NetSuite using authentication from a third-party IdP. This feature allows users who have logged in to an external application to go directly to NetSuite. Users do not need to log in separately to NetSuite, because authentication from the same IdP is used for login to both the external application and NetSuite. A user who accesses NetSuite using SAML SSO is directed to their NetSuite Home page. NetSuite account administrators can use role-based permissions in NetSuite to control which users have SAML SSO access to NetSuite.

Note: SAML single sign-on access to NetSuite UI honors any IP address rules for your company, or IP address restrictions for your employees, that you may have created in your NetSuite account. IP address rules or restrictions do not apply for SAML access to web stores or websites.

Task List for SAML SSO Set Up

Setting up SAML SSO requires some back-and-forth between NetSuite and the IdP of your choice.

1. In the NetSuite application, perform preliminary setup: enable the feature, create roles and assign SSO permissions, and assign users to the roles.
2. Using the IdP of your choice: create your NetSuite Service Provider (SP) configuration. The procedure varies depending on the IdP you choose to use.

Note: Some IDPs already have NetSuite listed among their out-of-the-box service providers, while others require that you configure the set up of NetSuite as new SAML Service Provider yourself.

3. In the NetSuite application, complete the SAML Setup page: create the configuration in your account for your IdP.

See the following sections for detailed information on each step:

- [Complete Preliminary Steps in NetSuite for SAML SSO](#)
- [Configure NetSuite with Your Identity Provider](#)
- [Complete the SAML Setup Page](#)

Complete Preliminary Steps in NetSuite for SAML SSO

To get started with SAML Single Sign-on (SSO), some preliminary setup steps must be completed in your NetSuite account. The feature must be enabled, and roles must be customized and SAML permissions added to those roles. SAML roles must be assigned to users.

The first step in setting up SAML SSO is to enable the feature, create roles and assign SSO permissions, and assign users to the roles.

See the following sections for detailed procedures:

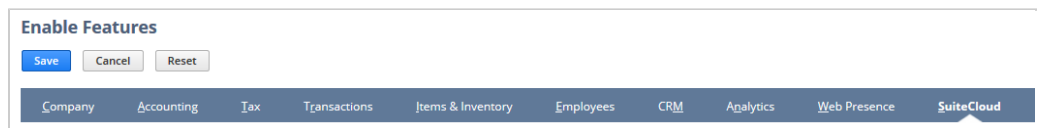
- [Enable the SAML Single Sign-on Feature](#)
- [Add SAML Single Sign-on Permissions to Roles](#)
- [Assign SAML Roles to Users](#)
- [Prepare to Provide NetSuite SP Metadata to Your IdP](#)

Enable the SAML Single Sign-on Feature

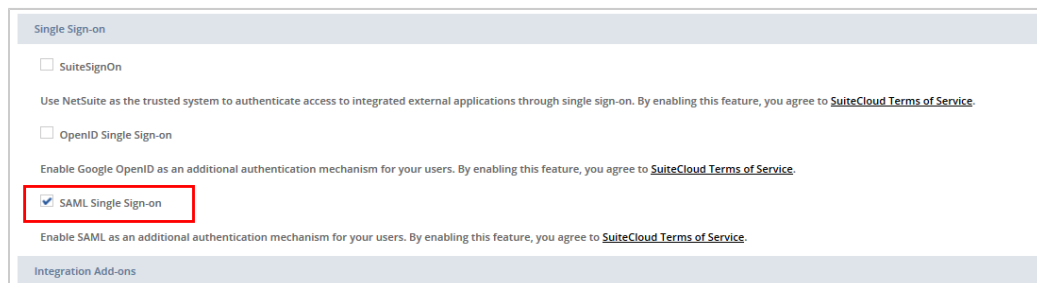
To complete the following procedure, you must be logged in to NetSuite with an Administrator role or in another role that has the Enable Features permission.

To enable the SAML Single Sign-on feature:


1. Go to Setup > Company > Enable Features and click the **SuiteCloud** subtab.



2. Scroll down to the Single Sign-on section, and check the **SAML Single Sign-on** box. Agree to the SuiteCloud Terms of Service when prompted.



3. Click **Save**.

 **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.

Add SAML Single Sign-on Permissions to Roles

You might want to customize a standard NetSuite role (or roles) to for use with SAML Single Sign-on (SSO) permissions. You can also add SAML SSO permissions to existing roles assigned to users that require this type of access. See the following sections for more information:

To complete the following procedure, you must be logged in to NetSuite with an Administrator role. If you need more detailed information about creating roles in NetSuite, see the help topic [Customizing or Creating NetSuite Roles](#).

To customize roles and add SAML permissions:

1. Go to Setup > Users/Roles > User Management > Manage Roles.
2. Choose a role and click **Customize**.
3. Create a unique and identifiable name for the role. For example, you could replace the word Customize in the role name with the word SAML.
4. Click the **Permissions** tab.
5. On the **Setup** subtab, select the appropriate SAML permission from the list, and click **Add**. There are two SAML permissions. Add one or both permissions to the role as appropriate. See [SAML SSO Permissions](#).
6. Click **Save**.


For more information about SAML permissions, see the following:

- [SAML SSO Permissions](#)
- [SAML SSO Access for Center Roles](#)
- [SAML SSO Permission Limitations](#)

SAML SSO Access for Center Roles

You can add the SAML Single Sign-on permission to customized versions of the following center roles: Customer Center, Employee Center, Vendor Center, Partner Center, and Advanced Partner Center. Center roles are different from other NetSuite roles in that you can only add a limited set of permissions to them.

To add the SAML Single Sign-on permission to a customized center role:

 **Important:** No special permission is required to grant a customer center role SAML access to a website. The SAML permission is enabled for all customer center users, after the SAML setup for the website is completed.

1. Go to Setup > Users/Roles > Manage Roles.
2. Click the **Edit** link for a customized center role or click the **Customize** link for a standard center role.

3. On the Role page, click the **Permissions** subtab.
4. On the **Setup** subtab, set the **Level** to **Full** for the SAML Single Sign-on permission.

SAML SSO Permissions

When the SAML Single Sign-on feature is enabled, the following permissions are available:

- **Set Up SAML Single Sign-on** - permits users other than NetSuite account administrators to view and edit the SAML Setup page. (The Administrator role already has this permission.)
- **SAML Single Sign-on** - requires users to log in to the NetSuite UI using SAML SSO. This permission must be explicitly assigned to a role. However, a user assigned this permission will not be able to log in to the NetSuite UI from the standard login page with their username and password.

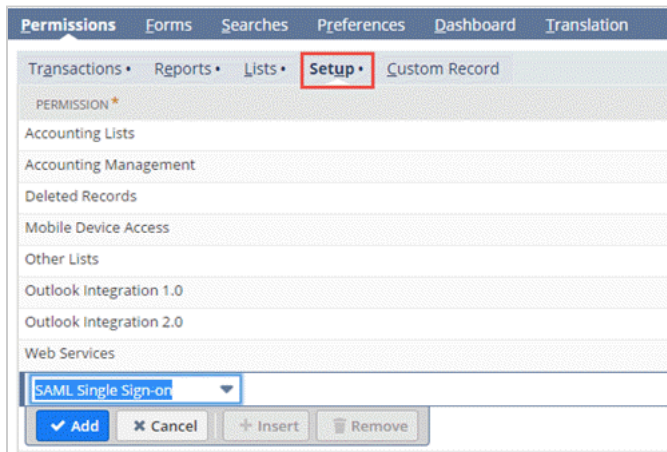
Important: No special permission is required to grant a customer center role SAML SSO access to a website. After the SAML setup for a website is completed, the SAML permission is automatically enabled for all customer center users.

Both of these permissions are Setup type permissions that support only a Full access level.

To provide SAML single sign-on access to users, the SAML Single Sign-on permission can be added to an existing role that is already assigned to users. Or, a new role can be created to which this permission can be added, and this new role can then be assigned to users.

Important: You cannot add SAML Single Sign-on permission to a role that has SuiteAnalytics Connect permission.

Permissions are added to roles on the Role record page, available at Setup > Users/Roles > Manage Roles.



Important: After the SAML Single Sign-on permission has been assigned to a role, there is a small delay before a user can use this role to log in using SAML single sign-on. This delay is related to caching; the new permission is not available until after the cache has timed out.

For more information about adding permissions to roles, see the help topic [Customizing or Creating NetSuite Roles](#).

SAML SSO Permission Limitations

SAML Single Sign-on roles and permissions have various limitations that are intended to prevent problems.

For example, the NetSuite account administrator role does not have SAML Single Sign-on permission and no user can log in using SAML single sign-on as an administrator. This limitation ensures that an administrator can always log in and resolve any problems that might occur with the third-party IdP setup or SAML access.

Another example of a limitation is that account administrators cannot add SAML Single Sign-on permission to a role that has SuiteAnalytics Connect permission. SAML access is not supported for SuiteAnalytics Connect.

Some limitations are intended to ensure that the account administrator has absolute responsibility for explicitly deciding who is allowed to access their NetSuite account using SAML Single Sign-on. The account administrator is deciding to trust the third-party IdP to authenticate and allow access to their NetSuite account. This is the reason for the following limitations:

- A user who has accessed NetSuite with a role that does not have SAML Single Sign-on permission cannot access any roles that do have SAML Single Sign-on permission.
- As of 2018.1, it is up to an account administrator to decide whether users should be locked in a single account. See [Account Attribute](#) for more information. (In previous releases, a user who accessed NetSuite through SAML Single Sign-on could not access any roles that belonged to a different NetSuite account. SAML Single Sign-on access was provided to only a single account.)

Some limitations are intended to ensure there are no conflicts resulting from having two different trust authorities (the third-party IdP and NetSuite) authenticating a single user. After SAML is enabled for certain roles in an account, NetSuite trusts the third-party identity provider. This is the reason behind the following limitations:

- A user with a role that has SAML Single Sign-on permission cannot log in directly to the NetSuite user interface using the standard NetSuite login page.
- A user who has accessed NetSuite through SAML Single Sign-on cannot access any roles that do not have SAML Single Sign-on permission. This prevents users from switching from a SAML role to a non-SAML role with greater privileges.
- Only one type of inbound single sign-on permission can be assigned to a specific role. If a role has SAML Single Sign-on permission, it cannot have OpenID Single Sign-on permission or be granted inbound single sign-on access through the NetSuite Inbound Single Sign-on feature.

Assign SAML Roles to Users

To complete the following procedure, you must be logged in to NetSuite with an Administrator role. If you need more detailed information, see the help topic [NetSuite Users Overview](#).

To assign a SAML Single Sign-on role to users:



Important: A user with a role that has SAML Single Sign-on permission cannot log in directly to the NetSuite user interface on the standard NetSuite login page with the SAML role.

The following procedure is for adding a role with the SAML Single Sign-on permission to users.

1. Find the appropriate entity record for the user. Go to List > Employee > Employee.
2. Click the name of the user.
3. Click the **Access** subtab. (For Customer, Partner, or Vendor entities, look for the Access subtab on the **System Information** subtab.)



Important: No special permission is required to grant a customer center role SAML access to a website. The SAML permission is enabled for all customer center users, after the SAML setup for the website is completed.

4. Click **Edit**.

5. Select your custom SAML Single Sign-on role from the list.
6. Click **Add**.
7. Click **Save**.

Prepare to Provide NetSuite SP Metadata to Your IdP

After the SAML Single Sign-on feature is enabled, NetSuite account administrators and users with the Set Up SAML Single Sign-on permission can view and edit the SAML Setup page in NetSuite. How you configure NetSuite as a Service Provider (SP) with the Identity Provider (IdP) of your choice depends on the IdP you have selected. To prepare for any eventuality, before you attempt to set up SAML with your IdP, you should gather some information from the SAML Setup page in NetSuite.

To copy the NetSuite SP metadata file and related URL:

The person responsible for configuring SAML access to NetSuite on the IdP side should perform the following steps.

1. Go to Setup > Integration > SAML Single Sign-on.
2. Copy the URL shown in the NetSuite Service Provider Metadata field, and save it where you can retrieve it when necessary.
3. Download the SP metadata file to your computer. Remember the location you save the file to.



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

SAML Setup

Submit

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://system. /saml2/sp.xml

LOGOUT LANDING PAGE *

☐ PRIMARY AUTHENTICATION METHOD

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

Permission Requirements

Users must have the SAML Single Sign-on permission in order to access NetSuite through SAML single sign-on. You can edit users' assigned roles to include this permission at Setup > Users/Roles > Manage Roles.

Submit

Note: If your account has been or is moved to a different data center location, information in the NetSuite SP metadata file will also change. You must provide the information in the SP metadata file to your IdP from the current location of your account.

Configure NetSuite with Your Identity Provider

It is not possible to provide detailed instructions for configuring NetSuite as a Service Provider (SP) with your Identity Provider (IdP). Refer to the documentation available from your IdP for configuring SAML access. However, see the following procedure for basic guidance on what must be accomplished to set up SAML access to NetSuite with your IdP. The exact steps will vary, depending on your IdP. The procedure will also vary depending on whether the NetSuite application is already configured by your IdP, or if you must create the NetSuite application yourself with your IdP.

Note: Your IdP could be a web application or an on-premises solution. The NetSuite application could already be included in their list of SP applications. The IdP might have a setup wizard or a manual to guide you through the process.

To configure SAML with your IdP:

1. Go to your IdP website or an on-premises administration console, and follow the application setup instructions from your IdP.

Note: You must create a new SP application for NetSuite. Refer to your IdP's documentation for directions on how to do this.

2. Provide the NetSuite Service Provider Metadata to your IdP by one of the following methods:
 - a. Upload the NetSuite SP metadata file, or:
 - b. Paste the URL for the NetSuite SP metadata file in the appropriate field with your IdP, or:
 - c. Manually configure SAML on the IdP side by copying information from specific fields in the NetSuite Service Provider Metadata file to the IdP.

**Your IdP
(website or
on-premises
console)**

From the NetSuite Service Provider Metadata file

SP Entity ID

Always refer to the NetSuite Service Provider Metadata file in your account.
Copy the SP entityID from the NetSuite Service Provider metadata file you downloaded from the SAML Setup page your account.
The SP entityID is shown in the first line of the file.

Assertion
Consumer
Service

Always refer to the NetSuite Service Provider Metadata file in your account.
Copy the URL from the NetSuite Service Provider metadata file you downloaded from the SAML Setup page in your account.
The value varies depending on the data center where your account is hosted, and on the account type. There are many AssertionConsumerService lines for Location in the file. Use the URL in your file that includes `isDefault="true"` in the line.

Single Logout
Service

Always refer to the NetSuite Service Provider Metadata file in your account.

Your IdP
(website or
on-premises
console)

From the NetSuite Service Provider Metadata file

Copy the URL from the NetSuite Service Provider metadata file you downloaded from the SAML Setup page in your account. The value varies depending on the data center where your account is hosted, and on the account type. Ensure you use a POST binding.



Important: Some sandboxes hosted in data centers in North America are still accessed from the `system.sandbox.netsuite.com` domain. If the sandbox has been upgraded to 2018.1, but has not been refreshed since January 11, 2018, the URLs in the file are different. The `entityID`, `AssertionConsumerService`, and `SingleLogoutService` values differ from those in sandboxes that have been refreshed since January 11, 2018. View the NetSuite Service Provider metadata file for your sandbox account to confirm the appropriate URLs.

3. Your IdP also has an IdP metadata configuration file. You must copy the URL for this file, or download the IdP metadata file. (Later, you must either enter the URL or upload the file into NetSuite on the SAML Setup page.)
4. With your IdP, you must assign (or provision) the NetSuite application to the SAML users in your account.

In many cases, the previous steps take care of all the information you need to provide to the IdP. For more information about signing assertions, encryption, and SAML attributes, see [IdP Metadata and SAML Attributes](#).

Complete the SAML Setup Page

When the SAML Single Sign-on feature is enabled, the SAML Setup page is available at Setup > Integration > SAML Single Sign-on, to NetSuite account administrators and to users with the Set Up SAML Single Sign-on permission. (For details about SAML Single Sign-on permissions, see [Add SAML Single Sign-on Permissions to Roles](#).)

Important: The URL link to the NetSuite Service Provider Metadata field in the following screenshot is obscured, because the URL varies depending on the account type and your data center location.

SAML Setup

Submit
Actions

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://system. /saml2/sp.xml

LOGOUT LANDING PAGE *

☐ PRIMARY AUTHENTICATION METHOD

Current Identity Provider

ENTITY ID

Current Identity Provider Metadata

Update Identity Provider

SAMLV2 IDENTITY PROVIDER METADATA

☒ INDICATE IDP METADATA URL

☐ UPLOAD IDP METADATA FILE

Permission Requirements

Users must have the SAML Single Sign-on permission in order to access NetSuite through SAML single sign-on. You can edit users' assigned roles to include this permission at Setup > Users/Roles > Manage Roles.

Submit
Actions

Note: If your account is moved to a different data center, information in the NetSuite SP metadata file will also change. You must provide the information in the SP metadata file to your IdP from the current location of your account. For instructions, see [Prepare to Provide NetSuite SP Metadata to Your IdP and Configure NetSuite with Your Identity Provider](#).

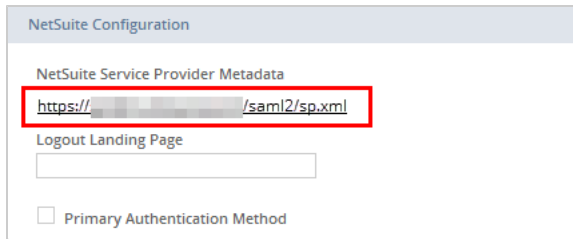
For details about completing the SAML Setup page, see:

- [Defining the NetSuite Configuration for SAML](#)
- [Update Identity Provider Information in NetSuite](#)

Note: To enable SAML access to a website (as opposed to the NetSuite application), you need to complete the SAML subtab of the Web Site Setup page. See the help topic [SAML Single Sign-on Access to Web Store](#).

Defining the NetSuite Configuration for SAML

To support SAML single sign-on access to NetSuite, you must define the following on the SAML Setup page:



NetSuite Configuration

NetSuite Service Provider Metadata

[https://\[redacted\]/saml2/sp.xml](https://[redacted]/saml2/sp.xml)

Logout Landing Page

☐ Primary Authentication Method

- Logout Landing Page - after logging in to NetSuite through SAML single sign-on, this is the URL for a page that users should be redirected to when they log out of NetSuite. An IdP Single Logout page can be specified for Single Logout to work.

Note: This solution is not part of the SAML 2.0 standard. There is no guarantee that this will work.

The Primary Authentication Method is optional.

- By default, the Primary Authentication Method option is disabled, so if SAML users click a link to access NetSuite when no active NetSuite session exists, they are redirected to the NetSuite login page. This redirect may cause issues for users who do not know their NetSuite credentials.
- If you enable the Primary Authentication Method option, users can be redirected to the external IdP login page. This redirect is available if:

- the user has already been logged in, the redirect occurs based on previous experience with NetSuite.
- the access link includes the NetSuite account ID set as the **c** or **compid** URL parameter, formatted like the following:

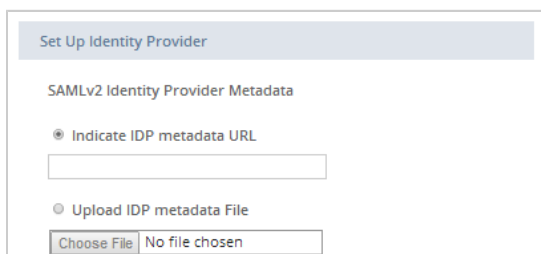
<https://system.netsuite.com/app/center/card.nl?c=<ACCOUNTID>>

Note: If the Primary Authentication method is enabled, and a user clicks a link containing the **c** or **compid** URL parameter, the user is redirected to the external IdP login page. The originally requested URL will be passed as a RelayState parameter, in accordance with the SAML 2.0 specification. This means that the IdP can direct the user back to the correct resource after authentication. If there is a live session for the IdP, the user will be directed back to the resource without being asked for credentials.

Set Up Your Identity Provider (IdP) in NetSuite

SAML single sign-on access to NetSuite requires that you specify an XML file that defines the identity provider to be used for authentication and includes required metadata for this identity provider. The format of this file must be aligned with SAML v2.0 specifications.

On the SAML Setup page, the IdP metadata file can be specified by entering a URL or by uploading the actual XML file. This is the information you gathered when you were setting up NetSuite with your IdP.



Set Up Identity Provider

SAMLv2 Identity Provider Metadata

☒ Indicate IDP metadata URL

☐ Upload IDP metadata File

No file chosen

You must do one of the following:

- Choose **Indicate IDP metadata URL** and enter the location URL of the metadata file.
- Or, choose the **Upload IDP metadata File** option and browse to locate the file.

IdP Metadata and SAML Attributes

See the following for more information about IdP metadata and specifying SAML attributes.

- [IdP Requirements](#)
- [NameID and Email Attributes](#)
- [SAML Response Example](#)

IdP Requirements


The Identity Provider metadata file should map required attributes between the identity provider and NetSuite, so that NetSuite can accept the identity provider's SAML assertions.

See the following:

- [Supported Encryption and Signature Options](#)
- [Mapping of SAML Attributes](#)
- [SAML Attribute Statements](#)
- [SAML Response Example](#)

Supported Encryption and Signature Options

At a minimum, NetSuite requires that an assertion be signed. Also, on the IdP side, an administrator can opt for several different levels of encryption.

 **Important:** Encryption of the entire SAML Response is not supported.

NetSuite supports the following levels of encryption:

- The whole assertion can be encrypted.
- All attributes and NameID can be encrypted.
- Only the attributes can be encrypted.
- Only the NameID can be encrypted.

Mapping of SAML Attributes

See the following table for a mapping of SAML attributes to NetSuite parameters, and whether they are required or optional.

SAML Attribute	NetSuite Parameter	Required or Optional
account	accountID	Optional, unless: <ul style="list-style-type: none"> ■ you are sending the role attribute. ■ you are sending the site attribute. ■ access to both non-customer center and customer center SAML roles is needed.

SAML Attribute	NetSuite Parameter	Required or Optional
		Sending the account attribute locks user access to a single account. See Account Attribute for more information.
role	role	Optional. See Role Attribute for more information.
site	site ID	Required for web store access. See Site Attribute for more information.
NameID or email	user email address	Required, must use NameID attribute or email attribute. If sending both attributes, they must be identical. See NameID and Email Attributes for more information.

SAML Attribute Statements

See the following sections for more information about SAML attributes.

- [Account Attribute](#)
- [Role Attribute](#)
- [Site Attribute](#)
- [NameID and Email Attributes](#)
 - [Supported NameID Formats](#)

Account Attribute

The `account` attribute is your NetSuite account ID. If you do not know your NetSuite account ID, a user with an administrator role can go to Setup > Company > Company Information to view the Account ID field. The `account` attribute is optional, unless:

- If you are sending the `role` attribute, then `account` is required.
- If you are sending the `site` attribute, then `account` is required.
- If users need access to both their non-customer center and customer center SAML roles, then `account` is required.



Important: If you send the account attribute, users are locked into a single company account, and will not be able to switch between multiple accounts that trust the same IdP.

Role Attribute

The ability to define a role ID is particularly useful if you have a Site Builder or SuiteCommerce Advanced website. It is not possible for a user to switch roles when logged in to a website. With the role attribute, you can define the SAML role to be used for login. The role defined in the assertion is treated as a default role.

The `role` attribute can be passed along with the SAML assertion as an additional attribute. If the `role` attribute is sent, the assertion must also include with the `account` attribute.

Site Attribute

Setting the `site` attribute (the site ID) is required for web store access. If you are sending the site attribute, you must also set the `account` attribute. If you do not know the site ID, see [Finding the Site ID Parameter](#).

NameID and Email Attributes

The user email is required. It must be provided either as the value in the `NameID` attribute or the `email` attribute.

Note: If using both the `NameID` and the `email` attributes, the values for these attributes must be identical.

Supported NameID Formats

The following formats are supported for the `NameID` attribute:

- `emailAddress`
- `transient`
- `unspecified`

Important: No matter which of these formats you choose to use, the `NameID` value must contain an email address.

The [SAML Response Example](#) illustrates the use of the `emailAddress` format for `NameID`.

The following line indicates use of the `unspecified` format:

```
<saml2:NameID Format="urn:oasis:names:tc:SAML:1.1:nameid-format:unspecified">jsmith@example.com</saml2:NameID>.
```

SAML Response Example

The following is an example of a SAML Response, showing parts of the SAML assertion element. If you do not provide the required attributes in your file, you receive error messages, for example: Email must be provided using NameID value or the email attribute.

The following example illustrates one way to provide these values:

```
...
<saml:Subject>
  <saml:NameID Format="urn:oasis:names:tc:SAML:1.1:nameid-format:emailAddress"
    SPNameQualifier="http://www.netsuite.com/sp"
    >jsmith@example.com</saml:NameID>
...
  <saml:AttributeStatement>
    <saml:Attribute Name="email">
      <saml:AttributeValue xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
        xsi:type="xs:string">jsmith@example.com</saml:AttributeValue>
      </saml:Attribute>
    <saml:Attribute Name="account">
      <saml:AttributeValue xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
        xsi:type="xs:string">123456</saml:AttributeValue>
      </saml:Attribute>
    <saml:Attribute Name="role">
      <saml:AttributeValue xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
        xsi:type="xs:string">1010</saml:AttributeValue>
      </saml:Attribute>
    <saml:Attribute Name="site">
      <saml:AttributeValue xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
        xsi:type="xs:string">1</saml:AttributeValue></saml:AttributeStatement>
```

```
</saml:AttributeStatement>
```

```
...
```

Update Identity Provider Information in NetSuite

After you have defined an identity provider for SAML Single Sign-on access, you can make changes as needed to the identity provider configuration on the SAML Setup page.

- You can remove the current identity provider metadata without replacing it with another identity provider, by clicking the Delete IDP Configuration button, under Actions. This action does not delete NetSuite Configuration information on this page.

SAML Setup

Submit Actions

Delete IDP Configuration

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](https://[redacted]/saml2/sp.xml)

Logout Landing Page
<https://system.netsuite.com>

☒ Primary Authentication Method

Note: For information on viewing or removing the identity provider metadata for SAML access to web stores, see the help topic [SAML Single Sign-on Access to Web Store](#).

- You can change the identity provider by uploading or entering a URL for an XML file that contains the metadata for a different identity provider.

☒ Primary Authentication Method

Current Identity Provider

Entity ID
<https://ib2.idp-example.com:1081/openss>

Current Identity Provider Metadata

Update Identity Provider

SAMLv2 Identity Provider Metadata

☒ Indicate IDP metadata URL

☐ Upload IDP metadata File
Browse... No file selected.

Permission Requirements
Users must have the SAML Single Sign-on permission in order to access NetSuite through SAML single sign-on. You can edit users' assigned roles to include this permission at Setup > Users/Roles > Manage Roles.

Submit Actions

Interactions with NetSuite Using SAML

SP-initiated and IdP-initiated Flows

There are two possible flows in the SAML 2.0 standard: SP-initiated and IdP-initiated. NetSuite supports both types of flows.

To trigger an SP-initiated flow:

- SAML must be set as a primary authentication method, or:
- A user should have a browsing history using SAML, and a deep link should be used to trigger the flow.

See the description of the Primary Authentication Method in [Defining the NetSuite Configuration for SAML](#).

Single Logout (SLO)

NetSuite has limited support for Single Logout (SLO) functionality. IdP-initiated SLO is supported for the NetSuite UI. The following is not supported:

- IdP-initiated SLO is **not** supported for SuiteCommerce web stores.
- SP-initiated SLO is **not** supported for the NetSuite UI or for SuiteCommerce webstores.



Note: This 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.

SAML SSO in Multiple NetSuite Account Types

If you are using SAML Single Sign-on (SSO) in more than one account type, be aware of the following information.

Set Up and Configure SAML SSO in More Than One Account

The Shared IdP feature in 2018.1 introduces the possibility to trust the same IdP from multiple NetSuite accounts.

This list details four important changes as a result of this Shared IdP feature:

1. There is no longer a unique constraint on the IdP entity ID in NetSuite.
2. Users can log in and switch between NetSuite accounts trusting the same IdP.
3. Administrators are no longer required to create independent service provider (SP) configurations on the IdP side for every NetSuite account.

4. Only one NetSuite SP configuration is required, which removes problems that may have been encountered due to IdPs requiring unique SP entity IDs.

You can use the same IdP metadata file for all your NetSuite account types (for example, production, sandbox, and Release Preview accounts).

- For accounts accessed from the NetSuite domain (`system.netsuite.com`) the SP entityID is `http://www.netsuite.com/sp`.
Account types accessed from the NetSuite domain include production, development, test drives, Release Preview, and some sandbox accounts.
- For sandbox accounts that are accessed from the sandbox domain (`system.sandbox.netsuite.com`) the SP entityID is `http://www.netsuite.com/sp-SANDBOX`. If your sandbox is still accessed from the sandbox domain, you cannot use a single SP configuration for all your accounts. A separate SP configuration for sandbox is required in this case.



Important: Some IdPs require a unique entity ID for every SP. If this is the case with your IdP, follow the procedures below to so that there is a single SP configuration for all NetSuite accounts.

Update Existing SAML Configurations

To use Shared IdP feature in accounts where SAML was enabled and configured in NetSuite before 2018.1, you must update your configuration.



Important: The following procedure is only necessary for SAML configurations that existed prior to 2018.1.

To update your SAML configuration for a single IdP:

To begin using the Shared IdP feature in accounts where the SAML SSO feature is currently enabled, complete the following procedure.

1. Go to the SAML Setup page in your production account: Setup > Integration > SAML Single Sign-on.



Note: If the SAML SSO is for your webstore, go to the SAML subtab of the SSO subtab of the Web Site Setup page. There is an additional field to complete for website SAML setup, Landing Page after Login. See the help topic [SAML Single Sign-on Access to Web Store](#).

2. Click **Submit**.

This action saves your existing SAML configuration, and permits the sharing of the same IdP configuration between multiple accounts.

Enable SAML in Multiple NetSuite Account Types

The following procedure does not contain all the details for setting up and configuring SAML. For more details on each step, see the following topics:

- [Complete Preliminary Steps in NetSuite for SAML SSO](#)
- [Configure NetSuite with Your Identity Provider](#)
- [Complete the SAML Setup Page](#)

To set up and configure for a single IdP in multiple accounts:

To use the same IdP in multiple NetSuite account types (for example, your production account and a sandbox that is accessed from the system.netsuite.com domain), complete the following procedure. To access the SAML Setup page, you must be a NetSuite account administrator or a user with the Set Up SAML Single Sign-on permission.

1. Go to Setup > Integration > SAML Single Sign-on in one of your NetSuite accounts (production, sandbox, or Release Preview account.)
2. Click the link below the field labeled **NetSuite Service Provider Metadata** to download the SP metadata file.
3. Following the instructions provided by your IdP, create a single SP configuration, and upload the NetSuite SP metadata file.



Important: Do not specify an account parameter on the identity provider's side.

4. Download the IdP metadata from your IdP.
5. Upload the same IdP metadata file in all of the NetSuite accounts that you want to use with SAML.



Note: It is also possible to use the Shared IdP feature for multiple accounts of the same type. For example, you can use a single IdP for two or more production accounts, or several sandbox accounts (as long as the sandboxes are all accessed from the same domain).

NetSuite SAML Certificate References

The NetSuite SAML certificate is referenced in the NetSuite Service Provider metadata and in the SAML identity provider (IdP) metadata. This certificate is valid for a designated period of time, usually a number of years. As the certificate expiration date approaches, NetSuite will renew it. After the renewed certificate is available, the NetSuite service provider metadata file will be automatically updated to include data from the renewed certificate. The NetSuite SAML Setup page, at Setup > Integration > SAML Single-Sign-on, provides a link that can be clicked to view the contents of this file. Certificate references in IdP metadata may not be automatically updated. Account administrators will need to review certificate references in IdP metadata, and manually update them as necessary, to ensure they point to the renewed certificate. NetSuite Customer Support will provide advance notice of SAML certificate expiration to affected customers.



Note: For information about removing SAML access to NetSuite after the SAML Setup page has been completed, see [Remove SAML Access to NetSuite](#).

Remove SAML Access to NetSuite

There are multiple ways to remove SAML single sign-on access to NetSuite.

- Either of the following actions removes SAML access to NetSuite for a user or group of users in your account:
 - Removing the SAML Single Sign-on permission from the users' roles.
 - In extreme cases, editing the users' employee records in NetSuite to make them inactive.
- The following actions remove SAML access to NetSuite for all users in your account:

- Ensure that SAML roles are either inactivated, or SAML permissions are removed from the roles.
- Disabling the SAML Single Sign-on feature in NetSuite.



Note: You can remove identity provider metadata on the SAML Setup page. See [Update Identity Provider Information in NetSuite](#)

For information on viewing or removing the identity provider metadata for SAML access to web stores, see the help topic [SAML Single Sign-on Access to Web Store](#).

OpenID Single Sign-on

The OpenID Single Sign-on feature in NetSuite supports inbound single sign-on to NetSuite from G Suite (formerly known as Google Apps for Work). This feature allows users who have logged in to G Suite to go directly to NetSuite. Users do not need to log in separately to NetSuite, because their Google identity is used to access their NetSuite data. NetSuite account administrators can control OpenID access through role-based permissions in NetSuite.

Note: As an alternative to OpenID, SAML 2.0 is available for single sign-on access to NetSuite. G Suite supports SAML 2.0. For more information, see [SAML Single Sign-on](#). Also, the G Suite Administrator Help website provides documentation on setup and configuration of SAML SSO to NetSuite. Click the following to view Google's documentation: <https://support.google.com/a/answer/6194968?hl=en>. After the setup and configuration is completed, your users can use their G Suite credentials to access NetSuite.

The OpenID Single Sign-on feature in NetSuite uses Google's OpenID Connect (OAuth 2.0 for login). OpenID Connect is a federated authentication protocol that allows single sign-on from an identity provider to applications that are set up as relying parties (or service providers). Single sign-on allows users to log in a single time, to the identity provider, and then move to other applications without needing to log in again.

Note: OpenID Single Sign-on access to NetSuite is only supported for UI to UI integrations. It is not supported for access to a web store, or for access to NetSuite through web services. OpenID Single Sign-on access is not supported for customer center roles.

The following steps are required to implement OpenID access to a NetSuite account:

1. Your organization must have previously set up and registered a domain in G Suite (formerly known as Google Apps for Work). OpenID access to NetSuite is available from this registered domain only, not from a @gmail.com account.
2. An account administrator, or another user with the Setup OpenID Single Sign-on permission, must complete the following configuration tasks in NetSuite:
 - Enable the OpenID Single Sign-on feature.
 - Record the Google Apps domain name in NetSuite.
 - Assign the OpenID Single Sign-on permission to users who need direct access from G Suite to NetSuite.

For steps to complete these tasks, see [Setting Up Google Account Access to NetSuite](#).

3. To directly access NetSuite, users must enter a URL to link directly to the site, for example:


```
https://system.netsuite.com/app/common/integration/openidlogin.nl?partner=Google&domain=example.com
```

The first time users connect to NetSuite from G Suite, they are required to enter their NetSuite user name and password to create a mapping between their Google identity and NetSuite identity. After this initial mapping is completed, login to NetSuite usually occurs automatically whenever the user accesses the OpenID Single Sign-on URL.

Important: After the initial mapping is completed, there are certain situations when users are presented with the Choose Roles page, instead of automatic login to NetSuite. These situations include when no default role has been selected and when the user has more than one role with the OpenID Single Sign-on permission.

For more information about the login process, see [Logging in to NetSuite from G Suite](#).

Setting Up Google Account Access to NetSuite

Note: As an alternative to OpenID, SAML 2.0 is available for single sign-on access to NetSuite. G Suite supports SAML 2.0. For more information, see [SAML Single Sign-on](#). Also, the G Suite Administrator Help website provides documentation on setup and configuration of SAML SSO to NetSuite. Click the following to view Google's documentation: <https://support.google.com/a/answer/6194968?hl=en>. After the setup and configuration is completed, your users can use their G Suite credentials to access NetSuite.

Your organization must have previously set up and registered a domain in G Suite (formerly known as Google Apps for Work). The registered domain is required to use the OpenID Single Sign-on feature.

To allow users to access NetSuite directly from G Suite, a NetSuite account administrator needs to complete the following NetSuite configuration tasks:

1. Enable the OpenID Single Sign-on Feature
2. Record the Registered Domain Name in NetSuite
3. Assign the OpenID Single Sign-on Permission to Users

Enable the OpenID Single Sign-on Feature

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

To enable the OpenID Single Sign-on feature:

1. In NetSuite, go to Setup > Company > Enable Features.
2. On the **SuiteCloud** subtab, check the **OpenID Single Sign-on** box.
3. Review the **Terms of Service** and click **I Agree**.
4. Click **Save**.

After you save the Enable Features page, you need to go to Setup > Integration > OpenID Single Sign-on to record your Google Apps domain name.

Record the Registered Domain Name in NetSuite

Your organization must have previously set up and registered a domain in G Suite (formerly known as Google Apps for Work). You must record the domain name on the Google OpenID Setup page in NetSuite. This page is only available after the OpenID Single Sign-on feature has been enabled in your account.

Google OpenID Setup

[Submit](#)

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

Setup and Configuration

To use this feature, you must register your domain with Google Apps. If you do not have a domain registered with Google Apps, [please create and/or register a domain first.](#)

☒ Enable OpenID Single Sign-on

Google Apps Domain

Permission Requirements

Users must have the OpenID Single Sign-on permission in order to access NetSuite through OpenID single sign-on. You can edit users' assigned roles to include this permission at Setup > Users/Roles > Manage Roles.

To record the registered domain name in NetSuite:

1. Go to the Google OpenID Setup page at Setup > Integration > OpenID Single Sign-on.
2. In the **Google Apps Domain** field, enter your organization's domain name, and click **Submit**.
 - The domain name should be in the format <second level and lower level domains>.<top level domain>. For example: mydomain.com.
Do not include parts that designate a particular host server like "www."
 - A Google Apps domain can only be used by one NetSuite account. An error occurs if you enter a domain name already in use by another NetSuite account. This error may occur in the following cases:
 - You have incorrectly typed your domain name.
Review the name and correct as necessary.
 - Your organization has multiple NetSuite accounts, and has recorded the Google Apps domain for another of your accounts in error.
You need to modify that other account by either: recording a new domain for it, or disabling the OpenID Single Sign-on feature and clearing the domain for it. Then you can enter the domain for the correct account.
 - The domain name is in use by an account owned by another organization.
If you receive an error that the domain is already in use, and you have not recorded it for one of your accounts, contact NetSuite Customer Support for assistance.

After you have submitted the Google OpenID Setup page, the **Manage Roles** list page displays, so you can assign the OpenID Single Sign-on permission as necessary.

Assign the OpenID Single Sign-on Permission to Users

You can provide Google OpenID access to NetSuite users by giving them the OpenID Single Sign-on permission. Generally, NetSuite permissions are assigned on a per-role basis. Each user may have multiple roles. To access NetSuite from G Suite (formerly known as Google Apps for Work), a user must have at least one role with this permission.

If you want to ensure that a user accesses NetSuite ONLY through OpenID single sign-on, you can assign that user a single role that is designed as Single Sign-on Only. For details, see [Setting Up a Single Sign-on Only Role](#).



Important: To directly access NetSuite, users must enter a URL to link directly to the site, for example:

`https://system.netsuite.com/app/common/integration/openidlogin.nl?partner=Google&domain=example.com`

To add the OpenID Single Sign-on permission to a role:

1. Go to Setup > Users/Roles > Manage Roles.
2. Click the **Edit** link or **Customize** link next to the role where you want to add the permission.
3. On the Role page, **Permissions** subtab, **Setup** subtab, select **OpenID Single Sign-on** and click **Add**.



Important: You cannot add the OpenID Single Sign-on permission to customer center roles.

4. Click **Save**.
5. Repeat these steps for all roles to which you want to add this permission.

Note the following:

- The OpenID Single Sign-on permission is not provided by default to the Administrator role or the Full Access role, and cannot be added to these roles.
- Be aware that permissions can only be added to custom roles. To add a permission to a standard role, you need to create a customized version of it. For more information about NetSuite roles, see the help topic [NetSuite Roles Overview](#).
- If the Global Permissions feature is enabled in your account, you can assign the Google OpenID permission on a per-user basis, to apply across all of a user's roles. For information about this feature, see the help topic [Using the Global Permissions Feature](#).

OpenID Single Sign-on Access for Center Roles

You can add the OpenID Single Sign-on permission to customized versions of the following center roles: Employee Center, Vendor Center, Partner Center, Advanced Partner Center, NetSuite Support Center, and NetSuite Support Center (Basic). Center roles are different from other NetSuite roles in that you can only add a limited set of permissions to them.



Important: You cannot add the OpenID Single Sign-on permission to customer center roles.

To add the OpenID Single Sign-on permission to a custom center role:

1. Go to Setup > Users/Roles > Manage Roles.
2. Click the **Edit** link for a customized center role or click the **Customize** link for a standard center role.
3. On the Role page, go to the **Permissions** subtab.
4. On the **Setup** subtab, set the **Level** to **Full** for the **OpenID Single Sign-on** permission.

Assigning the Set Up OpenID Single Sign-on Permission

When the OpenID Single Sign-on feature is enabled, in addition to the OpenID Single Sign-on permission, the Set Up OpenID Single Sign-on permission is available to be assigned to NetSuite roles (not including center roles). This permission allows users other than NetSuite account administrators to view and edit the Google OpenID Setup page.

Note: Administrators already have this permission.

Logging in to NetSuite from G Suite

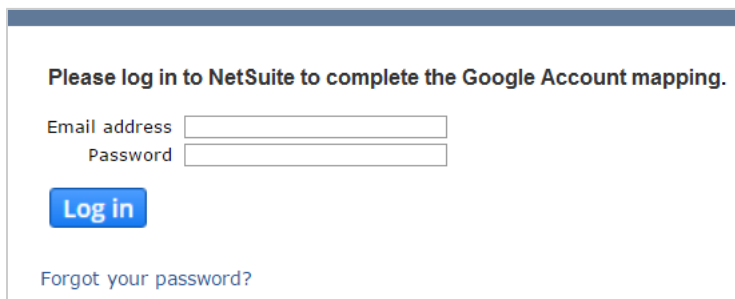
Note: As an alternative to OpenID, SAML 2.0 is available for single sign-on access to NetSuite. G Suite supports SAML 2.0. For more information, see [SAML Single Sign-on](#). Also, the G Suite Administrator Help website provides documentation on setup and configuration of SAML SSO to NetSuite. Click the following to view Google's documentation: <https://support.google.com/answer/6194968?hl=en>. After the setup and configuration is completed, your users can use their G Suite credentials to access NetSuite.

After an account administrator has completed the steps for [Setting Up Google Account Access to NetSuite](#), users can log in to G Suite, and then access NetSuite.

Important: To directly access NetSuite, users must enter a URL to link directly to the site, for example:

`https://system.netsuite.com/app/common/integration/openidlogin.nl?partner=Google&domain=example.com`

The first time a user accesses the OpenID Single Sign-on URL in a browser, a NetSuite login page displays, where the user needs to enter a NetSuite user name and password, to create a mapping between their Google identity and NetSuite identity.



The image shows a NetSuite login page with the heading "Please log in to NetSuite to complete the Google Account mapping." Below the heading are two input fields: "Email address" and "Password". Below these fields is a blue "Log in" button. At the bottom of the form is a link that says "Forgot your password?"

After this information is entered:

- If the OpenID Single Sign-on feature has not been enabled or a Google Apps domain name has not been recorded for the NetSuite account, an error occurs.
- If a user does not have any roles with the OpenID Single Sign-on permission, an error occurs.
- If a user has one role with this permission, the user is logged in with this role and taken to their NetSuite Home page.
- If a user has multiple roles with this permission, a Choose OpenID Role page is displayed, where the user can choose a role to use to log in, and optionally set a default role for OpenID access to NetSuite. The user is logged in with the chosen role and taken to their NetSuite home page.

Note: A user's default OpenID role can be different from their general NetSuite default role.

When a user logs into NetSuite the first time, the initial mapping is completed between NetSuite and Google Account. After the initial mapping is completed, login to NetSuite usually occurs automatically whenever the user accesses the OpenID Single Sign-on URL.



Important: After the initial mapping is completed, there are certain situations when users are presented with the Choose Roles page, instead of automatic login to NetSuite. These situations include: when no default role has been selected and when the user has more than one role with the OpenID Single Sign-on permission.

A user that has multiple roles with OpenID Single Sign-on permission can change among these roles during an OpenID session. If a user attempts to change to a role that does not have the OpenID Single Sign-on permission, a login page displays for the user to log in to a non-OpenID NetSuite session.

If a timeout occurs due to lack of use during a NetSuite OpenID session, the user is logged out, instead of the user interface being locked, as occurs in other NetSuite sessions.

Removing Google OpenID Access to NetSuite



Note: As an alternative to OpenID, SAML 2.0 is available for single sign-on access to NetSuite. G Suite supports SAML 2.0. For more information, see [SAML Single Sign-on](#). Also, the G Suite Administrator Help website provides documentation on setup and configuration of SAML SSO to NetSuite. Click the following to view Google's documentation: <https://support.google.com/a/answer/6194968?hl=en>. After the setup and configuration is completed, your users can use their G Suite credentials to access NetSuite.

There are multiple ways to remove Google OpenID access to NetSuite.

- Either of the following actions removes Google OpenID access to NetSuite for a user or group of users in your account:
 - Removing the OpenID Single Sign-on permission from the users' roles.
 - Editing the users' employee records in NetSuite to make them inactive.
- The following action removes Google OpenID access to NetSuite for all users in your account:
 - Disabling the OpenID Single Sign-on feature in NetSuite. (You can leave the domain name populated to preserve mappings, if you intend to enable the feature again in the future.)