SuiteAuthConnect

Authorize.net <> NetSuite Connector

Versions 2025.1 and up

Table of Contents

Overview

Guidelines for Use

DO NOT modify code if you want to have Cloud 1001, LLC provide support and future releases. Changing code will be overwritten by subsequent upgrades of the connector.

Installing the Software

NOTE

This software is not provided as a unified application - it is installed into your account using the SuiteCloud Development Framework provided by NetSuite(Oracle). The resulting installation is the same as if each component had been manually installed and configured in your account. This in no way impacts the availability or stability of the product - but it does mean if you should decide to uninstall the product at a future time it will be your responsibility to manually remove each component.

Prerequisites

Under Setup > Company > Enable Features >> SuiteCloud, bale the following:

- Custom Records
- Client SuiteScript
- Server SuiteScript
- SuiteFlow
- Token-based Authentication
- OAuth 2.0
- SuiteCloud Development Framework (SDF)
    - SDF and tokens may be disabled after installation if you want to but are required for updates to be applied

Enabling these components should have no visible impact to users in your system if they are not already enabled.

- Navigate to Customization > SuiteBundler > Search & Install Bundles
- Search for “SuiteCloud Development Integration” (bundle id 245955) and click on the name
- Walk through the installation steps
    - This is a NetSuite provided bundle that will enable the SuiteCloud Development Framework to act as an integration for pushing code into your account

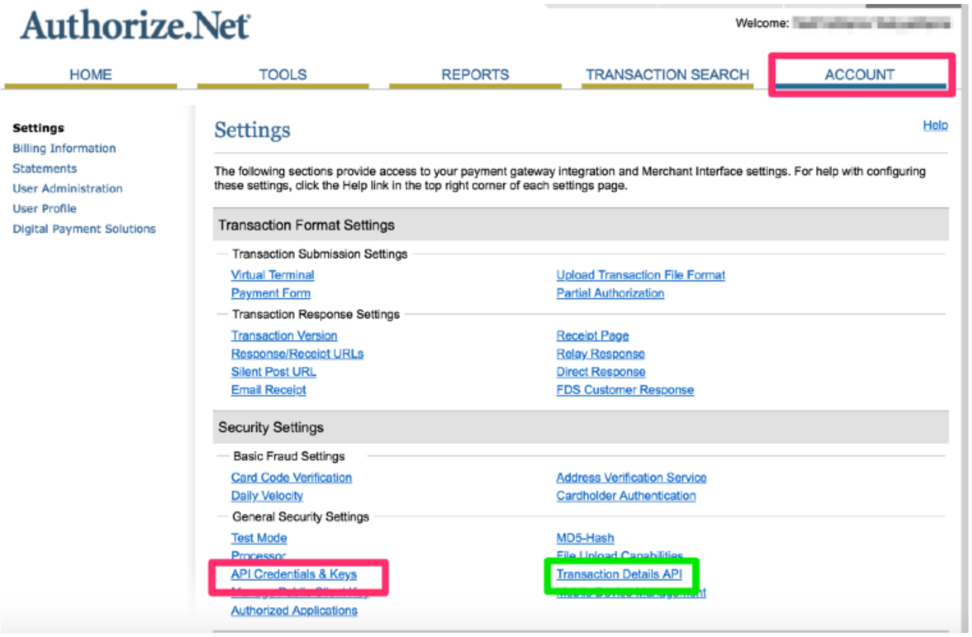
Installation

Install the code as a SDF Project following SuiteCloud Development Framework configuration in a supported IDE.

Post-Installation Configuration Steps

1. You will now need to log into your authorize.net gateway from a web browser so you can retrieve and enter you login and transaction key for your live and sandbox environments to allow the new software to connect to authorize.net. This information must be obtained from within your live and sandbox authorize.net accounts.

1. Log into the authorize.net environment (you will do this for both sandbox and production authorize.net)
    1. Production: https://account.authorize.net/
    2. Sandbox: https://sandbox.authorize.net/
        1. If you do not have a FREE authoriz.net SANDBOX account - you are STRONGLY encouraged to generate one now for testing - https://developer.authorize.net/hello_world/sandbox/
2. Generate a transaction key by navigating to the "Account" tab and selecting API Credentials & Keys
3. (inside the pink boxes)



Settings

The following sections provide access to your payment gateway integration and Merchant Interface settings. For help with configuring these settings, click the Help link in the top right corner of each settings page.

Transaction Format Settings

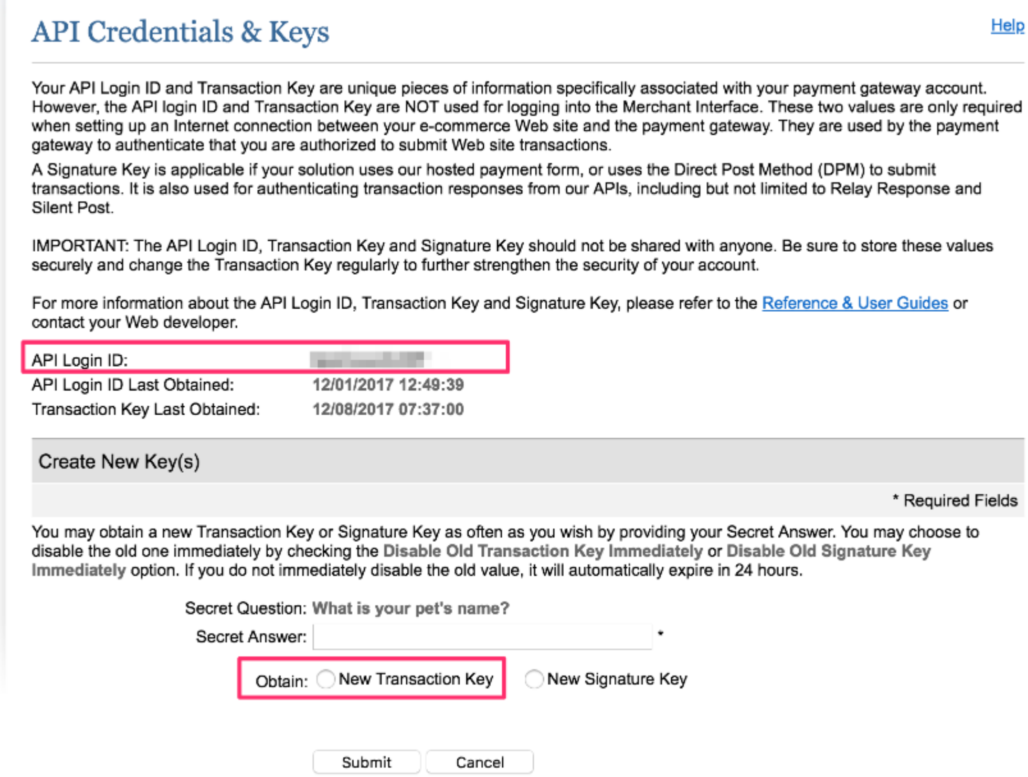
Transaction Submission Settings

Transaction Response Settings

Security Settings

Basic Fraud Settings

General Security Settings



API Credentials & Keys

Your API Login ID and Transaction Key are unique pieces of information specifically associated with your payment gateway account. However, the API login ID and Transaction Key are NOT used for logging into the Merchant Interface. These two values are only required when setting up an Internet connection between your e-commerce Web site and the payment gateway. They are used by the payment gateway to authenticate that you are authorized to submit Web site transactions.

A Signature Key is applicable if your solution uses our hosted payment form, or uses the Direct Post Method (DPM) to submit transactions. It is also used for authenticating transaction responses from our APIs, including but not limited to Relay Response and Silent Post.

IMPORTANT: The API Login ID, Transaction Key and Signature Key should not be shared with anyone. Be sure to store these values securely and change the Transaction Key regularly to further strengthen the security of your account.

For more information about the API Login ID, Transaction Key and Signature Key, please refer to the Reference & User Guides or contact your Web developer.

API Login ID:	[redacted]
API Login ID Last Obtained:	12/01/2017 12:49:39
Transaction Key Last Obtained:	12/08/2017 07:37:00

Create New Key(s)

* Required Fields

You may obtain a new Transaction Key or Signature Key as often as you wish by providing your Secret Answer. You may choose to disable the old one immediately by checking the **Disable Old Transaction Key Immediately** or **Disable Old Signature Key Immediately** option. If you do not immediately disable the old value, it will automatically expire in 24 hours.

Secret Question: **What is your pet's name?**

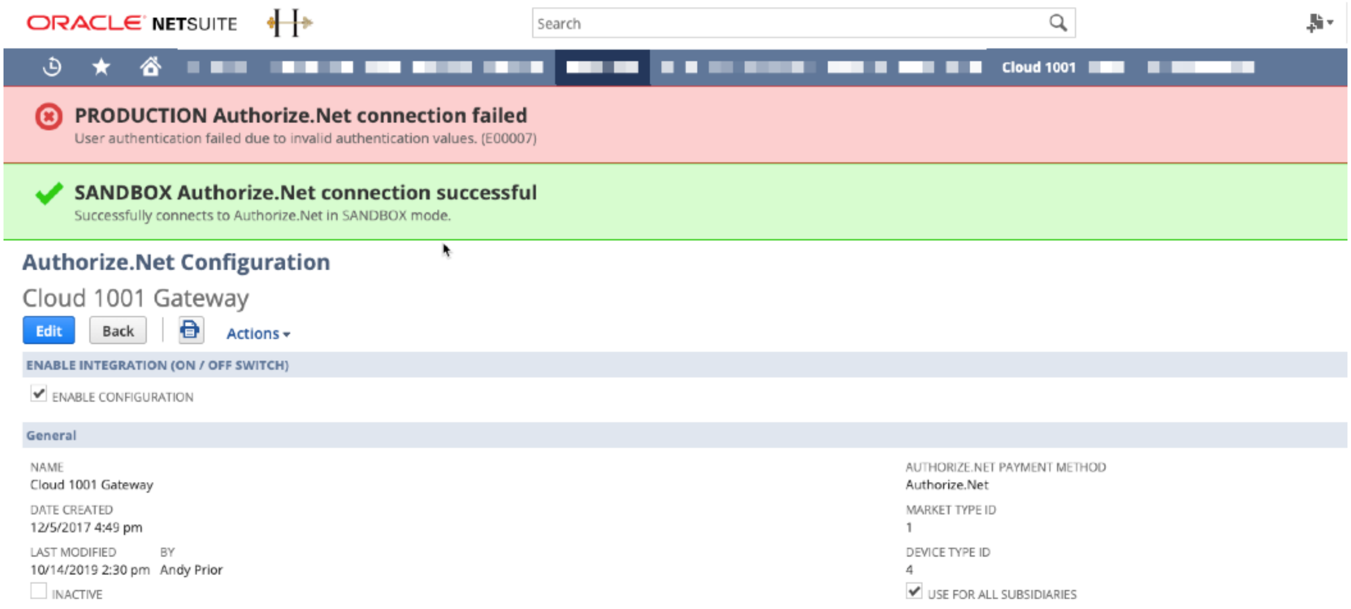
Secret Answer: __________ *

Obtain: ○ New Transaction Key ○ New Signature Key

2. Inside the API Credentials & Keys you will need to generate an API Login ID and Transaction Key
3. SAVE the keys to enter into NetSuite and be sure you remember which is which!
4. IN YOUR AUTHORIZE.NET SANDBOX - also make sure you enable the “live” mode in sandbox (if it’s not already enabled)… so the sandbox will act like “like” - but it will never really process a card.
5. Enable the Transactions Details API so you are configured for “settlement reporting” inside SuiteAuthConnect (the green box)

Entering Keys

1. Enter the API Login ID and the Transaction Key in the “Authorize.Net Configuration” record in NetSuite using corresponding fields of the same name - there are 2 sections “Production” and “Sandbox”
    1. You will need to do this for BOTH the production and sandbox authorize.net credentials - they will be different! (if you got your free sandbox)
2. DO NOT click the “Is Live” box in the NetSuite configuration under PRODUCTION until you are ready to start sending your credit card transactions to your production account! You should always test first
    1. It is important to note - even if “Is Live” is checked in a NON-PRODUCTION NetSuite environment (like a sandbox or preview release) - the connector will NEVER send transactions to your live Authorize.Net instance. ONLY a PRODUCTION account with “Is Live” checked will talk to your PRODUCTION Authorize.Net account.
3. After you enter your keys you can save your record. The record will test the keys you entered and let you know if there are any issues (this screenshot shows an issue with the production keys)



PRODUCTION Authorize.Net connection failed
User authentication failed due to invalid authentication values. (E00007)

SANDBOX Authorize.Net connection successful
Successfully connects to Authorize.Net in SANDBOX mode.

Authorize.Net Configuration
Cloud 1001 Gateway

ENABLE INTEGRATION (ON / OFF SWITCH)
☑ ENABLE CONFIGURATION

General

NAME
Cloud 1001 Gateway

DATE CREATED
12/5/2017 4:49 pm

LAST MODIFIED BY
10/14/2019 2:30 pm Andy Prior

□ INACTIVE

AUTHORIZE.NET PAYMENT METHOD
Authorize.Net

MARKET TYPE ID
1

DEVICE TYPE ID
4

☑ USE FOR ALL SUBSIDIARIES

OPTIONAL - External Integration

Many users of NetSuite opt to have non-NetSuite commerce - this section allows you to complete the transaction in NetSuite using authorize.net (including fund capture from your ecommerce AUTHORIZATION and any subsequent refunds, all using authorize.net)
It is important to note this assumes you are already using a 3rd party application to move the transactions from an external website into NetSuite and you must have some knowledge of the field mapping. You will be required here to enter the fieldId of the transaction ID that comes in from the external webstore (i.e. - if in Magento the order ID is MAGE123456789, whatever field that is mapped to in NetSuite is required)
You will also have to ADD to your field mapping in your webstore <> NetSuite integration the authorization string that is from the authorization event in your external web store and place that into the "TranId / RefId" field on the sales order that is provided with this (fieldId : "custbody_authnet_refid")

4. Checking "ALLOW EXTERNAL AUTH EVENTS" sets the stage for logic to look for a value in the "custbody_authnet_refid" field and in the field you identify in the next step
5. In the "EXTERNAL TRANSACTION FIELD ID", enter the fieldId that contains the mapped value of the identifier of the transaction in your external eCommerce System'
6. Checking the "VALIDATE AUTHORIZE.NET STATUS ON IMPORT" will ask the connector to CHECK each new transaction that is integrated to ensure the authorization value is valid. This is useful if you have multiple payment gateways mapped in or have difficulty mapping just certain transactions. And example might be a 3rd party connector is moving orders in from your Shopify store - and you support both authorize.net and PayPal in your store. If the 3rd party connector is incorrectly also sending PayPal transactions into the Authorize.Net fields, this will validate that and flag the order as invalid. It's also useful to just be extra sure before an item is sent for fulfillment that the authorization that is in NetSuite is a valid one.

OPTIONAL - Logging

The entire solution is PCI compliant and whole contained within NetSuite and authorize.net**.**
You DO have the ability to BREAK that compliance and see many of the data elements as they flow between the systems. Use at your own risk!
RECOMMENDATION- Do not change logging options unless instructed to do so in a support case

SAVING

Last Step - save the configuration record and all the changes you made are put into effect. This will again review your api keys and display a message if they are invalid.

Uninstalling

All custom records, custom lists, custom transaction fields, custom scripts and deployments must be manually removed from your account. There is no uninstall option provided by the NetSuite SuiteCloud Development Framework

Support

This is Open Sourced Software freely available and licensed in a way that allows modifications. This software is provided AS IS with no GUARANTEES of any sort. Assistance with installation, configuration and debugging is available from Cloud 1001, LLC on a Time and Materials basis if needed