Cardinal Centinel™ *for Merchants*

CARDINAL CENTINEL™

.Net Thin Client Installation Guide

CARDINAL COMMERCE

Acknowledgements

CardinalCommerce Corporation acknowledges with gratitude the contribution of its associates who developed the Cardinal Payment Authentication Platform.

Trademark Information

Contact Information

CardinalCommerce Corporation
6119 Heisley Rd.
Mentor, OH 44060
USA
www.cardinalcommerce.com

TABLE OF CONTENTS

1 Overview

This guide provides installation procedures, usage instructions and error codes for the Centinel .Net Thin Client. Also included is a description of the .Net samples provided and how those can be used in conjunction with the Centinel Test system.

2 Installation Procedures

Before you install:

Please ensure the following software or environments are present before installing the Cardinal Centinel Thin Client:

1. Microsoft .NET Framework Version 1.1 Redistributable Package. The .NET Framework version 1.1 redistributable package includes everything you need to run applications developed using the .NET Framework.

   Note: Although .NET Version 1.1 is the minimum requirement, the Cardinal Centinel Thin Client is compatible with Version 2.0.
2. Merchant Web Site/Shopping Cart software.

Installation Steps:

1. Extract the Centinel Thin Client archive file to a temp directory. (C:\tmp\centinel)
2. The archive file contains the file CMPCDotNet.dll. Copy this file from the C:\tmp\centinel\lib directory to the /bin directory of your commerce website.

At this point you have successfully installed the Cardinal Centinel Thin Client on the Merchant's web server. If the Merchant is running the web store from multiple web servers, the installation process must be repeated for each server.

3 .Net Thin Client API

Request Object - CentinelRequest

Object used for sending data to the Centinel MAPS.

Method	Description
Add	Add data values to the name value pairs used to construct XML Messages. **Usage :** object.Add(string name, string value) **Parameters :** name - name of the parameter value - value of the parameter **Returns :** None
SendHTTP	Accepts HTTP and HTTPS URLs to the Centinel MAPS server. This method will marshal and send all name/value pairs that have been added using the "Add Method". The second parameter is the timeout threshold (ms). This parameter sets the time limit that the server to server communication will await a response before throwing an exception. The recommended timeout setting is 10000 (10 seconds). **Usage :** object.SendHTTP(string transactionURL, int timeout) **Parameters :** transactionURL - fully qualified transaction URL timeout - timeout value (ms) **Returns :** CentinelResponse object
SendProxyHTTP	Accepts HTTP and HTTPS URLs to the Centinel MAPS server. Allows Proxy Server configuration information to be defined and used during transaction processing. This method will marshal and send all name/value pairs that have been added using the "Add Method". The second parameter is the timeout threshold (ms). This parameter sets the time limit that the server to server communication will await a response before throwing an exception. The recommended timeout setting is 10000 (10 seconds). **Usage :** object.SendHTTP(string transactionURL, int timeout, string proxyServerURL, string proxyServerPort, string proxyUsername, string proxyPassword) **Parameters :**

	transactionURL - fully qualified transaction URL timeout - timeout value (ms) proxyServerURL - fully qualified proxy server URL proxyServerPort - port of proxy server proxyUsername - credential username proxyPassword - credential password **Note:** If a username and password is not required by the proxy server, simply pass empty strings for each of these parameters. **Returns :** CentinelResponse object
GetUnparsedRequest	Retrieve the string value representing the XML equivalent of the name/value pairs that have been added to the CentinelRequest object. Useful for debugging purposes. **Usage :** object.GetUnparsedRequest() **Parameters :** None **Returns :** String value representing the XML response

Response Object - CentinelResponse

Method	Description
GetValue	Returns the value for a given named element returned on the response message. **Usage :** object.GetValue(string name) **Parameters :** name - name of the parameter **Returns :** String value of the parameter
GetUnparsedResponse	Returns the entire XML response message. Useful for debugging purposes. **Usage :** object.GetUnparsedResponse() **Parameters :**

	None **Returns :** String value representing the XML response

4 Lookup Message

Step 1: Reference the Thin Client

```
<%@ import Namespace="CardinalCommerce" %>
<%
    Dim ccRequest As New CentinelRequest()
    Dim ccResponse As New CentinelResponse()
%>
```

Step 2: Building the Message

```
<%
    ccRequest.add("Version", "1.7")
    ccRequest.add("MsgType", "cmpi_lookup")
    ccRequest.add("MerchantId", "123123")
    ccRequest.add("ProcessorId", "234234")
    ccRequest.add("TransactionPwd", "Passw0rd")
    ccRequest.add("TransactionType", "C")
    ccRequest.add("CardNumber", "4000000000000002")
    ccRequest.add("CardExpMonth","05")
    ccRequest.add("CardExpyear","2009")
    ccRequest.add("OrderNumber", request("order_number"))
    ccRequest.add("OrderDesc", request("order_desc"))
    ccRequest.add("Amount", "9995")
    ccRequest.add("CurrencyCode", "840")
    ccRequest.add("UserAgent", Request.ServerVariables("HTTP_USER_AGENT"))
    ccRequest.add("Recurring", "Y")
    ccRequest.add("RecurringFrequency", request("recurring_frequency"))
    ccRequest.add("RecurringEnd", request("recurring_end"))
%>
```

Step 3: Sending the Message to Centinel MAPS

```
<%
    ccResponse = ccRequest.sendHTTP(https://centineltest.cardinalcommerce.com/maps/txns.asp,
10000)
%>
```

Step 4: Checking the Message Results

```
<%
    errorNo = ccResponse.getValue("ErrorNo")
    errorDesc = ccResponse.getValue("ErrorDesc")
    enrolled = ccResponse.getValue("Enrolled")
    payload = ccResponse.getValue("Payload")
    acsurl = ccResponse.getValue("ACSUrl")
    transactionId = ccResponse.getValue("TransactionId")
%>
```

5 Authenticate Message

Step 1: Reference the Thin Client

```
<%@ import Namespace="CardinalCommerce" %>
<%
Dim ccRequest As New CentinelRequest()
Dim ccResponse As New CentinelResponse()
%>
```

Step 2: Building the Message

```
<%
ccRequest.add("Version", "1.7")
ccRequest.add("MsgType", "cmpi_authenticate")
ccRequest.add("MerchantId", "123123")
ccRequest.add("ProcessorId", "234234")
ccRequest.add("TransactionPwd", "Passw0rd")
ccRequest.add("TransactionType", "C")
ccRequest.add("TransactionId", "MDgyMTk2NDQyNTc4ow9ws+CVfg")
ccRequest.add("PAResPayload", pares)
%>
```

Step 3: Sending the Message to Centinel MAPS

```
<%
ccResponse = ccRequest.sendHTTP(https://centinel.cardinalcommerce.com/maps/txns.asp, 10000)
%>
```

Step 4: Checking the Message Results

```
<%
errorNo = ccResponse.getValue("ErrorNo")
errorDesc = ccResponse.getValue("ErrorDesc")
cavv = ccResponse.getValue("Cavv")
xid = ccResponse.getValue("Xid")
paresstatus = ccResponse.getValue("PAResStatus")
signature = ccResponse.getValue("SignatureVerification")
eciflag = ccResponse.getValue("EciFlag")
%>
```

6 Integration Samples

The Centinel Thin Client archive file also includes integration samples.

Copy the Samples directory from the extract location (C:\tmp\centinel) to a location on the web server. Before the samples can be used a few minor configuration values must be updated in CentinelConfig.aspx file based on your web server environment and assigned Centinel Merchant information.

Configuration Overview

Integration samples consists of the following files that outline an integration template for processing the Payer Authentication, PayPal, SECURE-eBill, and Bill Me Later transactions.

Integration Samples Template

File Name	Description
CentinelConfig.aspx	This configuration file centralizes all Centinel related configurables. These values are required to be defined to enable the samples for work properly.
CentinelUtility.aspx	This file has general utilities used for Thin Client Integrations.
ccMenu.aspx	Page creates the menu header for integration samples.
ccLookup.aspx	Simulate the checkout process of a ecommerce website.
ccProcessor.aspx	This file handles the sending and receiving of the cmpi_lookup message to MAPS.
ccLaunch.aspx	This page is used to POST the transaction request to External System. The External System will in turn display the authentication/payment information to the consumer within this location.
ccFrame.aspx	This page is used to display ccLaunch.aspx page within the framed window. The ccFrame.aspx page also references the ccHeader.aspx. The framed approach can be used to apply any website branding and merchant messaging around the authentication/payment window to inform the consumer.
ccHeader.aspx	Page creates the header for framed layout for the transaction processing.
ccVerifier.aspx	This page represents the TermUrl passed to the External System via the form POST on the ccLaunch.aspx page. The External System will post the results of the authentication(PARes)/payment to this page immediately following the consumer submitting their authentication/payment credentials. This page handles the sending and receiving of the cmpi_authenticate message to MAPS.
ccResults.aspx	This page is used to simply display the results of the transaction processing. This page simulates an order confirmation page.
ccTransfer.aspx	This file is used to POST the a PayPal Payment request to the Centinel Website for processing. Centinel will receive the request, construct the PayPal payment form and redirect the consumer to PayPal for payment.
ccNotification.aspx	This page receives the Centinel Notification Post and processes the cmpi_paypal_status message. Handles the sending and receiving of the cmpi_paypal_status message to MAPS.
ccBMLNotification.aspx	This page receives the Centinel Notification Post and processes the cmpi_bml_status message. Handles the sending and receiving of the

	cmpi_bml_status message to MAPS.
ccSEBNotification.aspx	This page receives the Centinel Notification Post and processes the cmpi_seb_status message. Handles the sending and receiving of the cmpi_seb_status message to MAPS.
ccCancelAgreement.aspx	This file handles the sending and receiving of the cmpi_paypal_cancel_preapproved_agreement message to MAPS.
ccInitiatePayment.aspx	This file handles the sending and receiving of the cmpi_paypal_preapproved_payment message to MAPS.
ccRefund.aspx	This file handles the sending and receiving of the cmpi_refund message to MAPS.
ccSearch.aspx	This file handles the sending and receiving of the cmpi_paypal_transaction_search message to MAPS.
ccSale.aspx	This file handles the sending and receiving of the cmpi_sale message to MAPS.
ccVoid.aspx	This file handles the sending and receiving of the cmpi_void message to MAPS.
ccAuthBridge.aspx	This file handles the sending and receiving of the cmpi_ab_lookup message to MAPS.
ccAuthorize.aspx	This file handles the sending and receiving of the cmpi_reauthorize message to MAPS.
ccReAuthorize.aspx	This file handles the sending and receiving of the cmpi_authorize message to MAPS.
ccCapture.aspx	This file handles the sending and receiving of the cmpi_capture message to MAPS.

7 .Net Thin Client Error Codes

Error Code	Description
9000	Unable to Communicate with MAPS Server
9010	Error Parsing XML Response
9020	The server name or address could not be resolved
9030	The URL does not use a recognized protocol
9040	HTTP(S) Request Timed Out or Invalid Timeout Specified