Installation CardControl SDK

Megan De Freitas - 2024-12-02 - SDK

Important Notice

After October 29, 2019, SalesPad will no longer be supporting CardControl. Additionally, the application will cease to be a PA-DSS validated solution as of this date, and therefore CardControl customers would no longer be PCI compliant.

Instead, SalesPad Desktop now offers built-in credit card processing via Nodus PayFabric. If you have questions or want more information on our credit card processing services, please contact your sales rep.

Overview

This document will detail the installation and usage of CardControl SDK. For more information on the CardControl SDK implementation, please see CardControl PA-DSS Implementation Guide. There are several items you must note relating to the installation before continuing. They are listed as follows:

- Two types of APIs are available with the CardControl SDK. There is both an application API, and a web API comprised of two web services (REST & SOAP).
    - The application API is used to display and utilize CardControl forms and controls to perform functionality. For example, SalesPad GP & Additions use the application API to display the CardControl transaction entry form.
    - The web services are used in many ways to integrate different applications or websites with CardControl.
- If using a .NET programming language, the SOAP service can be used by itself. If the language is not .NET based, it is recommended to use the REST application (which also requires the SOAP application to be installed and running).
- If using a .NET language and the SOAP application, the available functions can receive either:
    - An SDK provided object (a CustomerCreditCard, for example) and the desired serialization method OR An XML or JSON serialized string of a provided object.
- If using the REST application or a non-.NET language, a serialized string of the provided object will have to be provided and the serialization will have to be performed by the merchant's system.
- If using the web services, certain windows features (shown on the next page) must be enabled on the target server for IIS.

- From the implementation guide, page eight: The company database and web application server (server running the SOAP/REST application(s)) must be running on a different server than the publicly available web server and separated by an internal firewall.
- Two .zip files will be provided containing all necessary files for each web service.

SDK Server Installation and Configuration

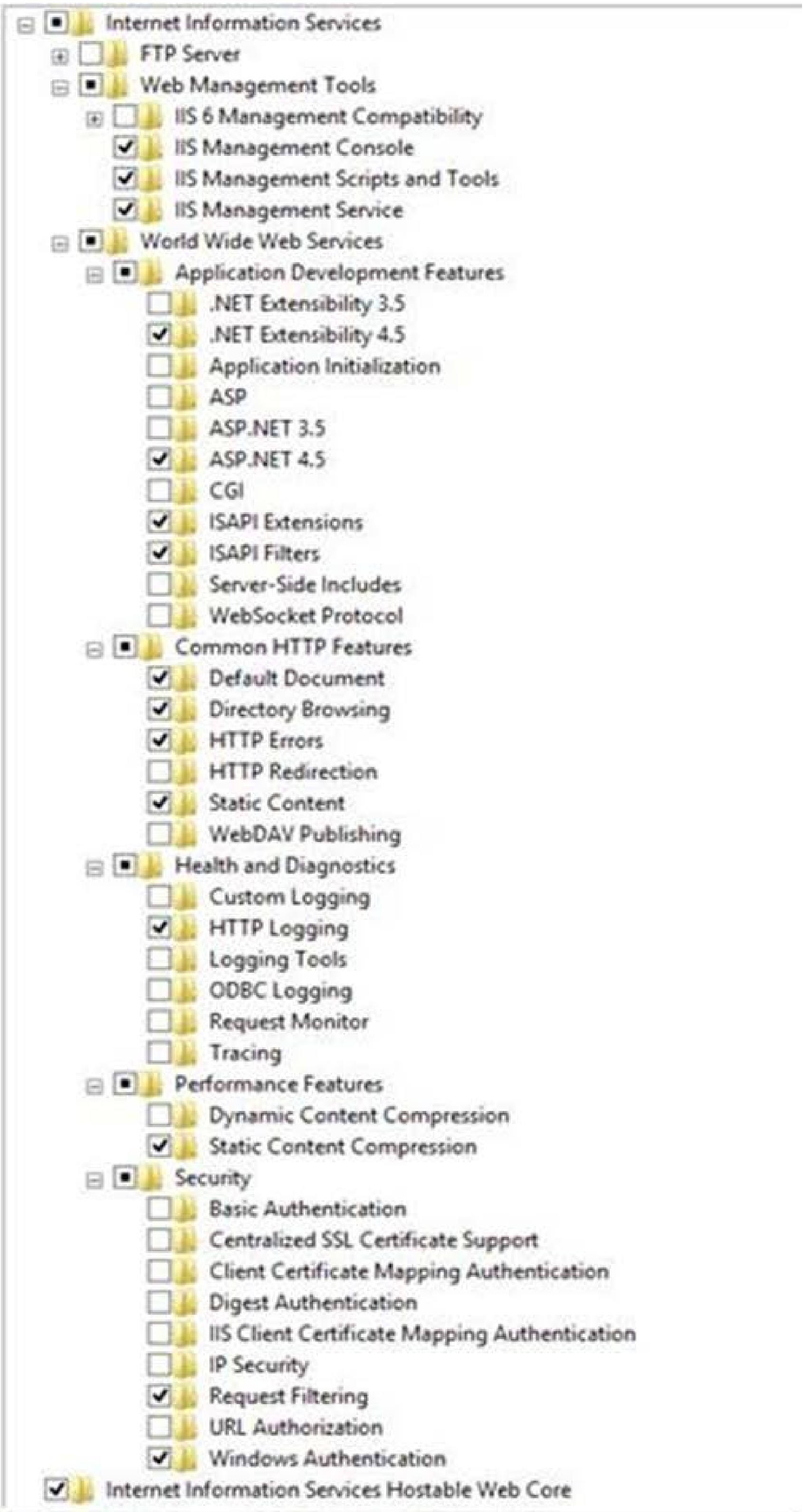
Please refer to the CardControl PA-DSS Implementation guide while installing and configuring the CardControl SDK. Unless the requirements in the implementation guide are fully fulfilled, the PA-DSS validation for the CardControl application will not be valid and the merchant's PCI inspection will have the CardControl application in scope.

- The implementation guide can be found here.
- Documentation regarding the functions and objects available for use in the SDK is available here.
- Code examples to implement the API are available on the last pages of this document and within the documentation listed above.

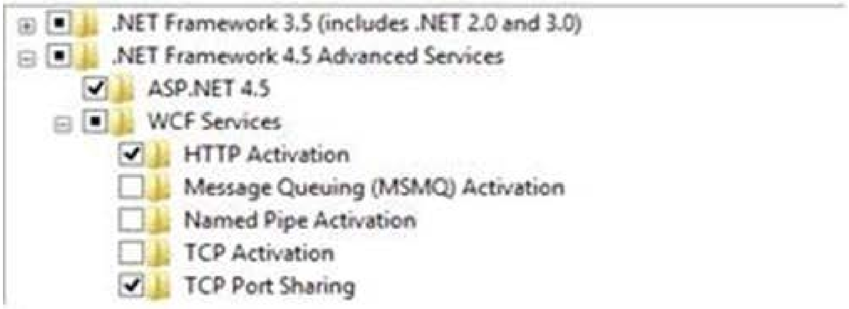
Configuring IIS

For IIS on Server 2008 R2, the Server features will have to be enabled in .NET 3.5, and THEN .NET 4.5 must be installed. This method assures that the settings in 3.5 will be the settings in 4.5.

Required features and services for SDK installation on IIS (this list may appear differently on various operating systems):



- ⊟ ■ Internet Information Services
  - ⊞ □ FTP Server
  - ⊟ ■ Web Management Tools
    - ⊞ □ IIS 6 Management Compatibility
    - ☑ IIS Management Console
    - ☑ IIS Management Scripts and Tools
    - ☑ IIS Management Service
  - ⊟ ■ World Wide Web Services
    - ⊟ ■ Application Development Features
      - □ .NET Extensibility 3.5
      - ☑ .NET Extensibility 4.5
      - □ Application Initialization
      - □ ASP
      - □ ASP.NET 3.5
      - ☑ ASP.NET 4.5
      - □ CGI
      - ☑ ISAPI Extensions
      - ☑ ISAPI Filters
      - □ Server-Side Includes
      - □ WebSocket Protocol
    - ⊟ ■ Common HTTP Features
      - ☑ Default Document
      - ☑ Directory Browsing
      - ☑ HTTP Errors
      - □ HTTP Redirection
      - ☑ Static Content
      - □ WebDAV Publishing
    - ⊟ ■ Health and Diagnostics
      - □ Custom Logging
      - ☑ HTTP Logging
      - □ Logging Tools
      - □ ODBC Logging
      - □ Request Monitor
      - □ Tracing
    - ⊟ ■ Performance Features
      - □ Dynamic Content Compression
      - ☑ Static Content Compression
    - ⊟ ■ Security
      - □ Basic Authentication
      - □ Centralized SSL Certificate Support
      - □ Client Certificate Mapping Authentication
      - □ Digest Authentication
      - □ IIS Client Certificate Mapping Authentication
      - □ IP Security
      - ☑ Request Filtering
      - □ URL Authorization
      - ☑ Windows Authentication
  - ☑ Internet Information Services Hostable Web Core

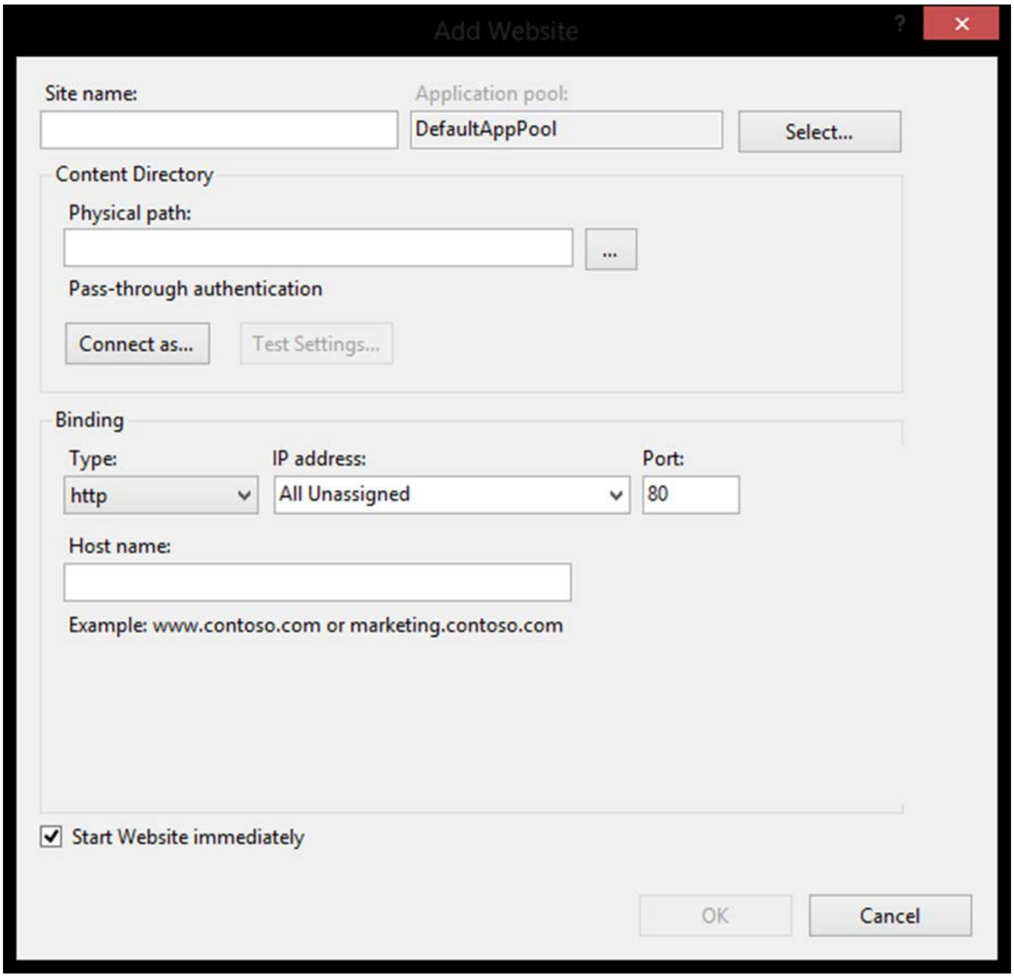


- ⊞ ■ .NET Framework 3.5 (includes .NET 2.0 and 3.0)
- ⊟ ■ .NET Framework 4.5 Advanced Services
  - ☑ ASP.NET 4.5
  - ⊟ ■ WCF Services
    - ☑ HTTP Activation
    - □ Message Queuing (MSMQ) Activation
    - □ Named Pipe Activation
    - □ TCP Activation
    - ☑ TCP Port Sharing

Adding and Configuring the API Website to IIS

On the app server, perform the following steps:

1. Open the IIS Manager application.
2. Add a new Site



Site name:

Application pool: DefaultAppPool

Physical path:

Type: http

IP address: All Unassigned

Port: 80

Host name:

☑ Start Website immediately

The following form will be displayed:

Site Name - This can be set to any value defined by the merchant.

Physical Path - This can be set to any value, but a typical path would be C:\inetpub\wwwroot\

Type - must be set to HTTPS.

IP address - This can be set to any available value.

Port - This can be set to the default value, or if this server is hosting another application that is already using the port, any preferred value.

Host Name - A verified SSL certificate is required for this. The host name must be covered by the domain name associated with the SSL certificate. This host name must also be set up in DNS at some location, probably locally. This host name must not be publicly accessible. We do not support installing or creating SSL certificates.

Example

For the SSL certificate domain name cavallo.com a valid hostname would be cardcontrol.cavallo.com.

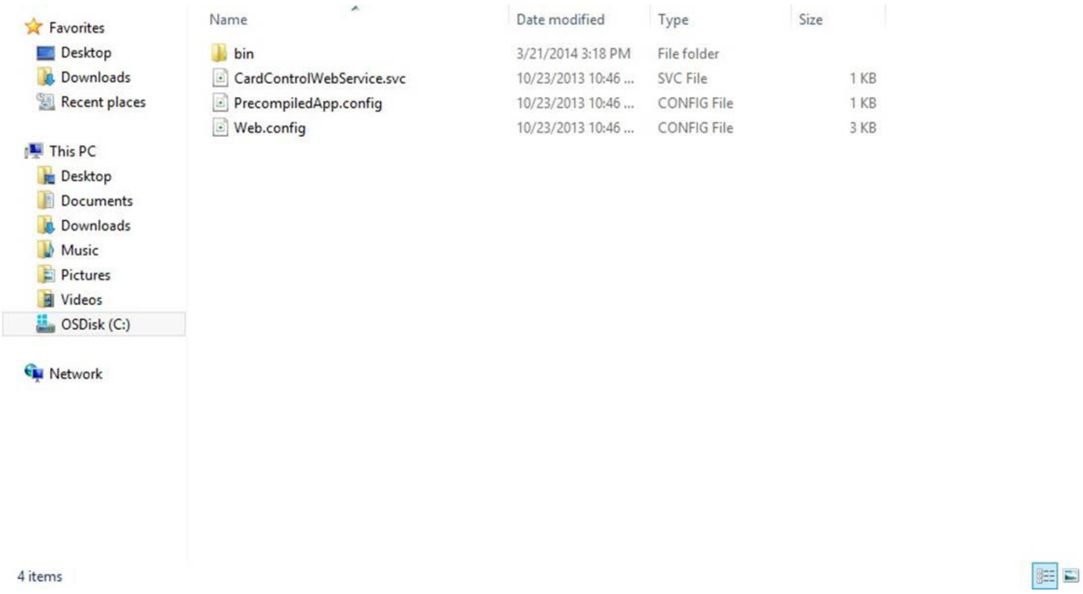
1. After the site has been added, click **Default Document** and set the highest priority document to be the .scv file included in the unzipped application folder.
2. SSL Settings: The Require SSL must be checked.
3. If using both SOAP and REST, each should be added as a separate application under the site. It is convenient to specify a physical path as a sub-folder of the site’s physical path.
4. Application Settings: The database connection string and username can be configured for the SOAP application. The username should be something similar to “CardControl API”. In this form, the REST application should be configured to have the correct URL to the SOAP application. These values can also be configured by directly modifying the web.config file in each application folder.

Important Notes

- Enable 32-Bit Application (viewable under the correct Application Pool): Must be set to true.
- .NET Framework Version: The version selected must be the version that reflects the .NET 4.5 installation.

Unzipped Application Files

The following files are available in the included .zip for the SOAP service. The REST service is very similar.



Name	Date modified	Type	Size
bin	3/21/2014 3:18 PM	File folder	
CardControlWebService.svc	10/23/2013 10:46 ...	SVC File	1 KB
PrecompiledApp.config	10/23/2013 10:46 ...	CONFIG File	1 KB
Web.config	10/23/2013 10:46 ...	CONFIG File	3 KB

- CardControlWebServic.scv – This is the file that should be configured to be the default document of the site.
- Web.config – This file contains parameters that should be modified accordingly for whichever service is being setup.
    - For the SOAP service, a valid SQL database connection string must be provided.
    - The SQL user used in the database connection string must have the SQL

permissions to read from the Dynamics database, or whichever database contains the Site Name/ID information if using other products.

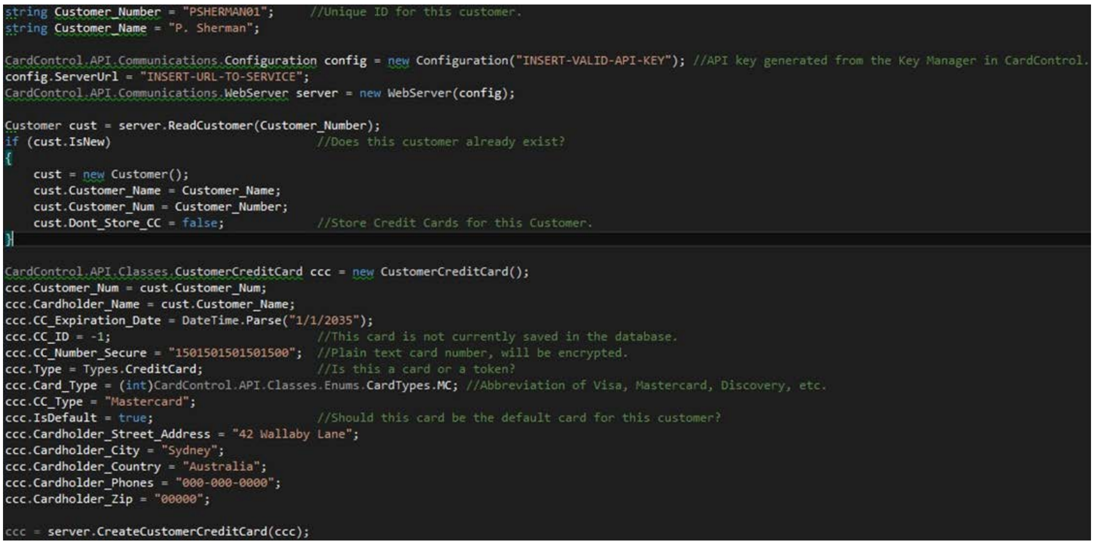
- Bin – This folder contains all required DLLs for the respective application.

Code Sample

Below is an example in C# in using the API. There are a few requirements for each call to the API. Two header values must be specified:

- Header: APIKey, Value: The API Key generated from the CardControl 2.0 API Key Manager (needs to be enabled in security).
- Header: Content-Type, Value: application\xml OR application\json.

Creating a customer and credit card using the SOAP service directly:



```
string Customer_Number = "PSHERMAN01";     //Unique ID for this customer.
string Customer_Name = "P. Sherman";

CardControl.API.Communications.Configuration config = new Configuration("INSERT-VALID-API-KEY"); //API key generated from the Key Manager in CardControl.
config.ServerUrl = "INSERT-URL-TO-SERVICE";
CardControl.API.Communications.WebServer server = new WebServer(config);

Customer cust = server.ReadCustomer(Customer_Number);
if (cust.IsNew)                            //Does this customer already exist?
{
    cust = new Customer();
    cust.Customer_Name = Customer_Name;
    cust.Customer_Num = Customer_Number;
    cust.Dont_Store_CC = false;            //Store Credit Cards for this Customer.
}

CardControl.API.Classes.CustomerCreditCard ccc = new CustomerCreditCard();
ccc.Customer_Num = cust.Customer_Num;
ccc.Cardholder_Name = cust.Customer_Name;
ccc.CC_Expiration_Date = DateTime.Parse("1/1/2035");
ccc.CC_ID = -1;                            //This card is not currently saved in the database.
ccc.CC_Number_Secure = "1501501501501500"; //Plain text card number, will be encrypted.
ccc.Type = Types.CreditCard;               //Is this a card or a token?
ccc.Card_Type = (int)CardControl.API.Classes.Enums.CardTypes.MC; //Abbreviation of Visa, Mastercard, Discovery, etc.
ccc.CC_Type = "Mastercard";
ccc.IsDefault = true;                      //Should this card be the default card for this customer?
ccc.Cardholder_Street_Address = "42 Wallaby Lane";
ccc.Cardholder_City = "Sydney";
ccc.Cardholder_Country = "Australia";
ccc.Cardholder_Phones = "000-000-0000";
ccc.Cardholder_Zip = "00000";

ccc = server.CreateCustomerCreditCard(ccc);
```

In this example, the CC_ID = -1 indicates that the card is not currently stored. This value will be populated with the new ID of the card if function is successful and it will be viewable in the returned card.