PKI Cert Creation via Good Control: Reference Implementation

Last updated: Thursday, July 27, 2017
Version: GC 4.0.xx.yy

Contents

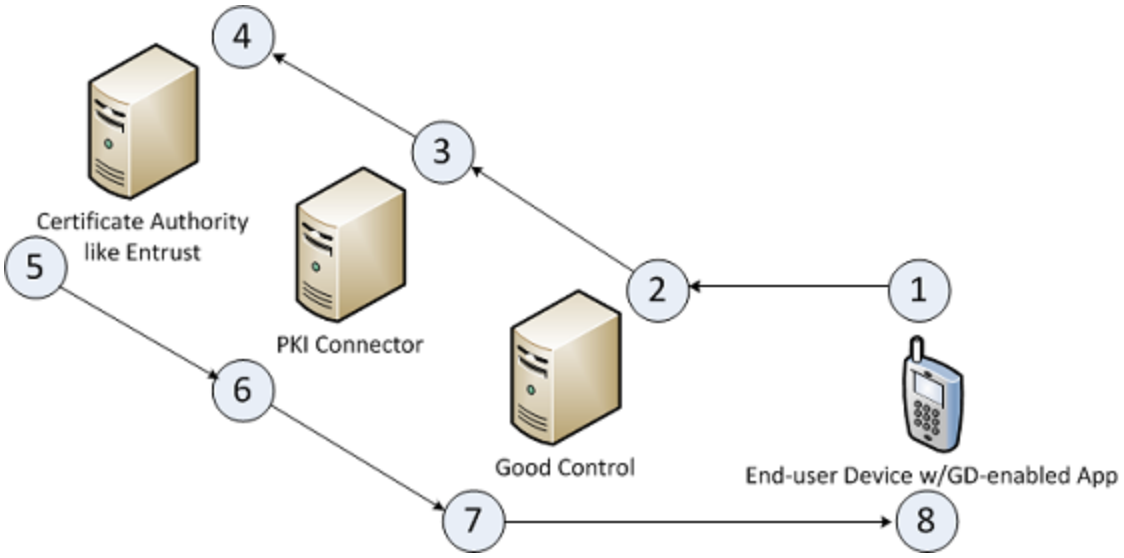
Revision history

Date	Description
2017-07-17	Updated to include details about renewing certificates. See Implementing certificate renewal
2017-03-07	Corrected link in Download the implementation to the downloadbale reference implementation
2017-01-27	The PKI Connector infrastructure is not available in the Good Cloud configuration. It requires an on-premise Good Control.server. See Required software and setup .
2016-12-19	Version numbers updated for latest release; no content changes.
2016-04-27	First publicly issued

Overview

This BlackBerry Dynamics reference implementation is one possible way to implement Good's User Certificate Management Protocol for programatic SSL/TLS certificate management. It illustrates how to obtain PKCS 12 certificates from the Entrust certificate authority (CA) via this set of Java programs and web services, which is called a *PKI connector*.

This set of programs works in conjunction with Good Control.



1. A BlackBerry Dynamics-enabled application makes a request to Good Control to authenticate the end-user.
2. The end-user's request causes Good Control to contact the PKI connector (the compiled reference implementation running on an application server):
   - On Good Control's **Certificates** > **Certificate Definitions** page, the IT administrator has created a definition that points to an application server that houses the compiled program (a "web archive", or war file) from this reference implementation.
   - In Good Control, the IT administrator has set the security policy **Allow use of client certificates**.
3. The program creates a Certificate Signing Request (CSR) and submits it to the Certificate Authority (CA). The reference implementation relies on Entrust certificate services, although any CA service can be used.
4. The CA issues a PKCS 12 certificate.
5. The CA sends the certificate data back to the program.
6. The program decrypts the certificate data and sends the data to Good Control.
7. Good Control sends the certificate to the end-user's device.
8. The BlackBerry Dynamics-enabled application installs the certificate on the device. Depending on a setting in GC, the end-user is prompted for a password to protect the certificate.

Intended audience skills

This guide and reference implementation are for Java programmers familiar with SOAP, web services, application servers, Certificate Authorities (CAs), and SSL/TLS certificates for PKI. You should also be familiar with Good Control and BlackBerry Dynamics in general.

About BlackBerry Dynamics software version numbers

The cover of this document shows the base or major version number of the product, but not the full, exact version number (which includes "point releases"), which can change over time while the major version number remains the same. The document, however, is always current with the latest release.

Product	Version
Good Control	4.0.57.102
Good Proxy	4.0.57.88
BlackBerry Dynamics SDK for Universal Windows Platform	3.0.0.805
BlackBerry Dynamics Bindings for: • Xamarin.Android • Xamarin.iOS	• Android: 3.2.0.2338 • iOS: 3.2.0.2465

If in doubt about the exact version number of a product, check the BlackBerry Developer Network for the latest release.

Contacting support for application development

For assistance with difficulties developing a BlackBerry Dynamics-based application, there are several resources:

1. The developer support forums on the BlackBerry Developer Network at https://community.good.com/community/gdn/support
2. Use the MyAccount system at http://myaccount.blackberry.com

For the second option, include the following details:

- Platform: Android, iOS, macOS, Windows
- BlackBerry Dynamics SDK version number
- Name of IDE or other tools
- Any build logs that you think are useful to solve your difficulty

Required software and setup

The reference implementation is a set of Java programs and support files, packaged as a Maven project.

- Java 8, including the JDK and JRE. Java 7 can be used, but requires more set up than is discussed in this guide.
- Apache Maven 2.x.x or 3.x.x
- Apache Tomcat 6, 7, or 8, or your preferred application server

On-premise Good Control

The PKI Connector infrastructure is not available in the Good Cloud configuration. It requires the installation of the on-premise Good Control server.

Operating systems

The reference implementation can be built on Microsoft Windows, Linux systems, or macOS.

Outbound network connection

The implementation also relies on other software, which Maven will acquire from the Internet when you build the software.

Make sure your development machine has an outbound connection to the Internet.

Entrust account needed

To make use of this reference implementation, you need to have an account with Entrust to authenticate to Entrust's services.

See Entrust's web site for details: https://www.entrust.com/

Entrust will give you a username, password, service endpoint, and other details that you supply to the reference implementation before building.

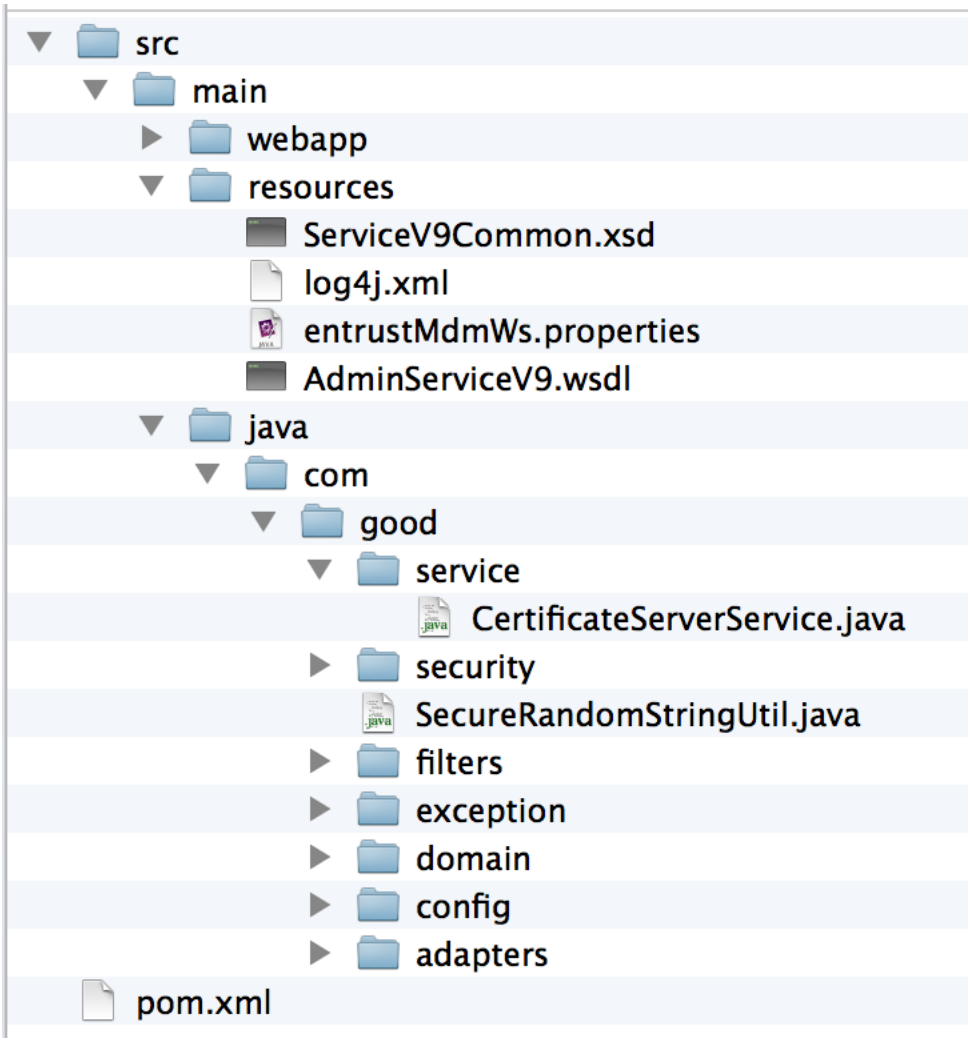
Download the implementation

The reference implementation source code is available in zip format on the BlackBerry Developer Network.at https://community.good.com/docs/DOC-7398.

1. Save the **`ga-ca-adapters.zip`** zipfile in a convenient location
2. Unzip it.

Directory structure

The high level structure of the directories is shown below:



The most important of these directories and files are pointed out in the remainder of this guide.

Build the software

The reference implementation is set up as a Maven project.

Add your Entrust details and other properties

Before you build, you need to add your own personal credentials (username and password) and other details from Entrust when you created an account with them.

1. For safety's sake, make a backup copy of the **`src/main/resources/entrustMdmWs.properties`** file.
2. Edit the original file.
3. Set the appropriate values for the variables listed below. You do not need to enclose the values in quotation marks.

Property	Optional/Required	Description
entrust.service.url	Required	URL of Entrust service endpoint. Obtain this value from Entrust.
entrust.login	Required	Your login to the Entrust account
entrust.password	Required	Base-64-encoded Entrust account password
entrust.digital.id.config.name	Required	Name of Digital Id Configuration, which is specific to Entrust and which you obtain from them.
request.userName	Required	The username to use in requests to Entrust service
request.userEmail	Required	The email address to use in requests to Entrust service
entrust.P12LatestKeyOnly=true	Optional	P12LatestKeyOnly boolean must be true to get uploaded PKCS 12 certificate's history.
entrust.iggroup	Optional	Can be used to create a Domain Name (DN) in the certificate for an individual Active Directory group, in conjunction with next two properties. Left blank, the implementation generates a random string to obtain a unique certificate.
entrust.deviceType	Optional	Any arbitrary string. Left blank, the implementation generates a random string to obtain a unique certificate.
entrust.deviceId	Optional	Any arbitrary string. Left blank, the implementation generates a random string to obtain a unique certificate.

4. Save the file.

Build with Maven

- Make sure you have Maven installed and in your path.
- Make sure your **JAVA_HOME** and **CLASSPATH** environment variables are correctly set.

To build the reference implementation:

1. Open a command window or shell.
2. Change directory to **gd-ca-adapters** top-level directory.
3. Start the build with this command:

   ```
   mvn clean install
   ```

4. Examine the output messages for any errors that might occur and correct them if necessary.

A successful build ends with the following message:

```
[INFO] BUILD SUCCESS
```

Check the built output in target directory

A successful build creates a web archive (war) file and associated files in the **gd-ca-adapters/target** subdirectory:

```
$ ls
classes                                maven-archiver
gd-ca-adapters-0.52-SNAPSHOT           maven-status
gd-ca-adapters-0.52-SNAPSHOT.war       surefire-reports
generated-sources                      test-classes
generated-test-sources
```

The war file needs to be deployed on an application server, the same server that you specify in GC's **Certificate Definitions** tab, which is discussed in Configure Good Control with your implementation .

Deploy in Tomcat or other app server

The details here focus on the popular Apache Tomcat application server, but other application servers can be used.

Recommendation: app server separate from Good Control

To maintain separation of concern, BlackBerry recommends that you do *not* deploy Tomcat or other application server on the same machine that is running Good Control.

Store client certificates in keystore for Tomcat

Tomcat needs a file location called the "keystore" to store the client certificates it receives from Entrust.

Tomcat also needs a file location called the "truststore" to store the certificates of servers it trusts, such as Entrust's server.

Using the **`keytool`** command, you should first create the **`keystore`** and the **`truststore`** files on disk as detailed in this link: https://docs.oracle.com/cd/E19509-01/820-3503/6nf1il6er/index.html

Make note of the paths to these files you create, so you can add the paths to the definition in Tomcat described in Add SSL connectors to Tomcat server.xml .

> **Important:** If you have chosen to run Tomcat on the same machine as Good Control (which is *not* recommended; see Recommendation: app server separate from Good Control) do *not* store the client certificates in Good Control's own keystore.

Import the Entrust server certificate

You need a copy of the Entrust servers' public certificate to put into Tomcat so the reference implementation can connect to Entrust.

- Run the following commands:

1. **`cd`** *`path_to_unzipped`***`/gd-ca-adapters/src/main/java/com/good/adapters/entrust/util`**
2. **`java GenerateSSLCertificate-NEED-TO-VERIFY-EXACT-CLASS-Cert.java`**
3. Copy the resulting file **`jssecacerts`** file to your Tomcat server's **`$JAVA_HOME/jre/lib/security`** directory.

Add SSL connectors to Tomcat server.xml

Add connectors to the **`$TOMCAT_DIR/conf/server.xml`** file to specify the port the implementation listens on and the paths to files that store certificates: the keystore and the truststore that you created in Store client certificates in keystore for Tomcat .

Note: The following is only an example of entries in **$TOMCAT_DIR/conf/server.xml**. Be sure to set the ports and paths to your deployment.

```
<Connector port="8090" protocol="HTTP/1.1"
       connectionTimeout="20000"
       redirectPort="8443" />

<Connector
       protocol="HTTP/1.1"
       port="8443" maxThreads="200"
       scheme="https" secure="true" SSLEnabled="true"
       keystoreFile="/c:/newcerts/foobar.jks" keystorePass="foobarpwd"
       truststoreFile="/c:/newcerts/cacerts.jks" truststorePass="cacertspassword"
       clientAuth="true" sslProtocol="TLS"/>
```

Example: Deploying the war file

Exact steps for deploying the war file depend on the application server software you are using and its configuration. For details about deploying with Tomcat, see https://tomcat.apache.org/tomcat-8.0-doc/deployer-howto.html.

The easiest way to deploy the built software with Tomcat is to copy the **gd-ca-adapters-0.52-SNAPSHOT.war** to the **webapps** directory of your application server, either manually or with a deployment manager program, such as comes with Apache Tomcat.

You might want to rename the war file to match whatever naming conventions your organization uses and depending on the look of the URL you will use to access it via Good Control. For instance, if you rename the file as **ROOT.war**, you can run it by navigating your browser to the root of the application server.

You need to know this URL to define in Good Control in Configure Good Control with your implementation .

Assumption: Tomcat is running the reference implementation in a war file named as shown in the first column on an example server name and port in a fictitious **BigCompany.com** domain. The URL to access the reference implementation is shown in the second column.

Example War File name, Host, and Port	Example URL to Run Reference Implementation and
• war File Name **ROOT.war** • Hostname **myAppServer** • Port 443	**https://myAppServer.BigCompany.com/**

Configure Good Control with your implementation

After you have built and deployed the reference implementation's WAR file to your application server, you need to configure Good Control to talk to it and to allow the use of client certificates on devices.

Setup certificate definition

To configure Good Control with the reference implementation:

1. Navigate to **Certificates** > **Certificate Definitions**.
2. Click **Add Definition**.
3. Enter a mnemonic name for this definition, such as "Entrust Adapter".
4. Click **Add**. You are presented with a screen of fields to configure.
5. Fill in the fields of the definition.

Field	Description
Server Address	The exact URL to your deployed war file, as detailed in Example: Deploying the war file . Use the **Test Connection** button to verify that a connection can be established.
Authenticate with username and password	If your application server relies on HTTP basic authentication, supply the username and password needed to connect to it. • Mutually exclusive with next setting. • Without modification, the reference implementation works with either setting.
Authenticate with client certificate	Upload a client certificate (PKCS 12 file) needed to authenticate to the application server. • Mutually exclusive with above setting. • Without modification, the reference implementation works with either setting.
Require user-entered password or OTP [one-time password]	Check this setting (default) if you want the end-user to have to set a password for the PKCS 12 certificate that is sent to his device.

6. Click **Save** to retain your changes or **Cancel** to discard them.

Allow client certificates

You need to set a security policy in Good Control that allows client certificates on devices.

These steps assume that the affected policy sets have already been associated with end users. If not, see the Good Control help topic "Modifying user accounts".

To enable client certificates in Good Control:

1. Determine the names of the affected policy sets. These are the policy sets associated with end users on whose devices you want to allow client certificates.
2. Navigate to **Policy Sets** > **edit a policy** > **Security Policies** tab.
3. Scroll to find the heading **Certificate Management**.
4. Check the setting **Allow use of client certificates**.
5. Click **Update** to save your changes or **Cancel** to discard them.
6. Repeat these steps for all affected policy sets.

Test your implementation

You can test connectivity to the running implementation war file by navigating your web browser to the following URL:

your_tomcat_url`/test`

where *your_tomcat_url* is the URL you devised in Example: Deploying the war file and the same **Server Address** you configured in Good Control in Configure Good Control with your implementation .

The returned result is:

```
It works.
```

Because the reference implementation relies on many different connected systems that are not feasible to duplicate in an end-to-end test environment, from the end-user device to the Entrust service, the only certain way to test is to test in production.

Before you advertise the service, try the reference implementation on one of your own devices.

Approaches to adapting this adapter

This section lists some of the more important points to keep in mind if you plan to modify this reference implementation to work with a CA other than Entrust.

Starting point: CertificateServerService.Java

Perhaps the best way to get started is to look at the following top-level program:

`src/main/java/com/good/service/CertificateServerService.java`

This program will lead you to the various touch points in the reference implementation that you need to account for in your own adapter.

GC's user certificate management protocol

The reference implementation adheres to the protocol and processes defined in Good's User Certificate Management Protocol.

If you plan to adapt this reference implementation to your own needs, such as to work with a different CA, you must make sure your adaptation adheres to this protocol.

Generating stubs

The reference implementation relies on the following files from Entrust, which are included with the software:

- **`AdminServiceV9.wsdl`**
- **`ServiceV9Common.xsd`**

These definitions were used to generate program stubs with Axis 1.4 for the reference implementation.

Your chosen CA might have similar definitions that you can use for your own adaptation. You might also need to use a later version of Axis.

Property handling

The reference implementation relies on settings in **`src/main/resources/entrustMdmWs.properties`**. as described in Build the software .

You will need to set up a properties file specific to your chosen CA.

Format of certificate data from CA

The reference implementation receives a byte stream of the unencoded certificate data from Entrust.

The format of the certificate depends on the CA. You might need to modify for your adapter to process data in a different format.

Implementing certificate renewal

The reference implementation as delivered does not include the logic necessary to work with the certificate renewal feature.

The necessary design aspects of certificate renewal are detailed in BlackBerry's User Certificate Management Protocol. Essentially, your PKI Connector must include a function to return values that indicate the capabilities of your connector. Those capabilities are as follows:

- getP12: New cert enrollment only
- getP12, renewCert: Bother new cert enrollment and certificate renewal

After you modify your PKI Connector and deploy it, you need to inform Good Control that the connector has new capabilities.

The latest version of Good Control includes an **Update connector capabilities** button (under **Certificates** tab) whereby you inform Good Control of your PKI connector's capabilities. The server makes a request to your connector to discover the capabilities based on the values you return.

BlackBerry Dynamics documentation

Category	Title	Description
Cross-platform	• *Getting Started Guide for Marketplace Partners* • *Good Control/Good Proxy Platform Overview for Administrators and Developers* • *Good Cloud Deployment*	Overviews of the BlackBerry Dynamics system
	• *Good Control Device and Application Management* • *Good Control Device Management Enrollment: Good Agent for iOS* • *Good Control Device Management Enrollment: Good Agent for Android*	Device and application management on Good Control, including app distribution, with client-side device enrollment details
Security	*BlackBerry Dynamics Security White Paper*	Description of the security aspects of BlackBerry Dynamics
	BlackBerry Dynamics and Fingerprint Authentication	Discussion of the implementation of BlackBerry security with fingerprint recognition systems: Apple Touch ID and Android Fingerprint
BlackBerry UEM	BlackBerry UEM Administration Guide	Approaches to administering the BlackBerry Unified Endpoint Manager
	Getting Started with BlackBerry UEM and BlackBerry Dynamics	Introductory material to administering the BlackBerry UEM with the BlackBerry Dynamics profile
Good Control	• *BlackBerry Secure Enterprise Planning Guide* • *BlackBerry Secure Servers Compatibility Matrix* • *BlackBerry Performance Calculator*	Guidelines and tools for planning your BlackBerry Secure Enterprise deployment
	Good Control/Good Proxy Server Preinstallation Checklist	Same checklist extracted from the installation guide below
	Good Control/Good Proxy Server	Details on installing Good Control, Good Proxy, and the GC

Category	Title	Description
	Installation	database
	Kerberos Constrained Delegation for Good Control	Configuration details for integrating the Kerberos authentication system with BlackBerry Dynamics
	Direct Connect for Good Control	Configuring BlackBerry Dynamics servers to securely access internal resources from the external Internet
	Good Control Easy Activation Overview	A look at the Easy Activation feature
	Good Control/Good Proxy Server Backup and Restore	Minimal steps for backing up and restoring the BlackBerry Dynamics system
	Good Control Online Help Good Control Server Property and Security Policy Reference	Printable copy of the GC console online help
	PKI Cert Creation via Good Control: Reference Implementation	A reference implementation in Java for creating end-user PKI certificates via Good Control and a Certificate Authority (CA)
	Good Control Cloud Online Help	Printable copy of the Cloud GC console online help
	Technical Brief: BlackBerry Dynamics Application Policies with XSD for app-policy XML	Description of formatting application policies for use in Good Control, with examples.
	Development Guide: Good Control Web Services	Programmatic interfaces on Good Control • Basic control and application management: SOAP over HTTPS. Documentation is in the WSDL files included with GC. • Device management: HTTP API (with JSON) for device management. Zipfile of API reference.
Software Development	*Developer Bootstrap: Good Control Essentials*	Bare minimum of information that a developer of BlackBerry Dynamics applications needs to get started with the Good Control server to test applications.
	BlackBerry Dynamics Shared Services Framework	Description of the BlackBerry Dynamics shared services framework for software developers
Android	• *Development Guide: BlackBerry Dynamics SDK for Android* • *API Reference for Android*	Working with the BlackBerry Dynamics SDK for Android and the essential reference for developers
iOS	• *Development Guide: BlackBerry Dynamics SDK for iOS*	Working with the BlackBerry Dynamics SDK for iOS and the essential reference for developers

Category	Title	Description
	• *API Reference for iOS*	
macOS	• *Development Guide: BlackBerry Dynamics SDK for macOS* • *API Reference for macOS*	Working with the BlackBerry Dynamics SDK for macOS and the essential reference for developers
Windows	• *Development Guide: BlackBerry Dynamics SDK for Universal Windows Platform (UWP)* • *API Reference for UWP*	Working with the BlackBerry Dynamics SDK for Universal Windows Platform (UWP) and the essential reference for developers
iOS, Android	*BlackBerry Launcher Library*	Source code and header files for implementing the popular BlackBerry Launcher interface
Cross-platform	*Development Guide: BlackBerry Dynamics SDK for Cordova for iOS and Android*	Working with the BlackBerry Dynamics SDK for Cordova plugins
	BlackBerry Dynamics Secure HTML5 Bundle Getting Started Guide for Developers	Working with the BlackBerry Dynamics SDK and the secure HTML5 bundle
	BlackBerry Dynamics Bindings for Xamarin.Android	Working with the BlackBerry Dynamics SDK and the Xamarin cross-platform integrated development environment For Xamarin.Android, no separate API reference is needed; see the standard BlackBerry Dynamics SDK *API Reference for Android*
	BlackBerry Dynamics Bindings for Xamarin.iOS and the *API Reference for Xamarin.iOS*	Working with the BlackBerry Dynamics SDK and the Xamarin cross-platform integrated development environment