User Guide: ServiceNow

Entrust Certificate Services

Support Hours: 8:00PM ET Sunday to 8:00PM ET Friday
Phone: 1-866-267-9297 (North America) 1-613-270-2680 (Rest of World)
# Table of Contents

**Introduction** ................................................................................................................................................. 3

**Prerequisites** .................................................................................................................................................. 3

1. Create ServiceNow developer account ................................................................. 4

2. Create an application ................................................................................................................................. 8

3. Create ECS SSL/TLS client auth profiles................................................................. 10

4. Configure ECS outbound SOAP service ............................................................. 20

5. Create certificate request application ................................................................. 28

6. Publish the application ............................................................................................... 46

7. Appendix A: Sample javascript for CSR submission ................................. 49

8. Appendix B: Sample javascript for certificate approval ........................ 51
Introduction

ServiceNow is a cloud service portal. This document describes how to configure ServiceNow to integrate with Entrust Certificate Services (ECS) SOAP API.

Prerequisites

Note: This procedure has been tested using a Geneva glide-geneva-08-25-2015__patch8-hotfix1-09-28-2016_09-28-2016_1638 instance.

For any ServiceNow application to communicate with ECS, it must be configured with ECS credentials and ECS user account. For detailed ECS account information, refer to ECS web service guide. The ECS credential, usually in PKCS#12 format, is used for ECS web service SSL/TLS mutual authentication; the ECS user account (user name and password) is used for ECS service (SOAP-based web service) API authorization.
1. Create ServiceNow developer account

This section is for the application developer to set up a development environment. For testing and production system, go section 3, Create ECS SSL/TLS client auth profiles.

1.1 Register as a developer

1.1.1 Go to https://developer.servicenow.com, click “REGISTER” to make a new Dev instance in ServiceNow.

1.1.2 Fill the information and "Submit".

1.1.3 Go to your email account, click the link to finish registration.

Just click on the link below to validate your address:

https://signon.service-now.com/ssoactivate.do?token=493a3ed3a8f22208d175f759a5736b6d8&email=huanqiu.wang2015%40gmail.com
1.2 Sign in to the new account

A few questions so we can maximize your experience...

* What best describes you?  
(choose the best answer)

- I am a current/potential partner
- I am a current/potential customer
- I am an independent developer
- Other

* What do you want to do?  
(choose all that apply)

- Learn about the platform
- Build an application for my company
- Sell an application on the Store
- Become a certified developer
- Other

*Country*  
Canada

*Company*  
My Company

Tell us more

Submit

1.3 Create a new instance

1.3.1 Once you’re signed into your ServiceNow developer account, click on “Request Instance”.

My Instance

You can request a ServiceNow instance free of charge. In order to retain the instance, you must have a minimum amount of activity or the instance will be reclaimed.

Request Instance
1.3.2 Select a version.

![ServiceNow Version Selection](image)

**Note:** This document assumes user is using ServiceNow Geneva version. There seems to be a problem for the latest ServiceNow Helsinki version loading the ECS soap message. Likely this is the ServiceNow issue. Test shows the integration script for ECS Soap call stills works in Helsinki, but all the service stub has to be manually configured.

1.3.3 Record user name "admin" and password; then click on “Manage Instance”.

![My Instance](image)
1.3.4 Record the URL. This is the URL to access your ServiceNow instance. Click on the URL and login with the admin password from the previous step.

My Instance

![My Instance](image1)

1.3.5 Once you are logged in, you should see Service Management screen within your ServiceNow instance.

![ServiceManagement](image2)
2. Create an application

2.1 You should be logged in using your instance URL, e.g. https://dev23313.service-now.com.

2.2 From the navigator pane, navigate to System Applications ◊ Applications; click "Applications".

2.3 In the content pane, click "New" button, on the top right side.

2.4 The application options are displayed. Click "Create" button to the right of the "Create custom application" option.
2.5 In the “Create Application” page, provide a name for your application in the “Name” field and then select “Create”.

2.6 Confirm creation of the new application in the popup window. Once the Application Creation is complete, click the cross mark to close the dialog box.
3. Create ECS SSL/TLS client auth profiles

The ECS Web Service API requires that the SOAP client connecting to the API authenticates using SSL client authentication. The ServiceNow platform requires a keystore in either the PKCS12 format or the Java JKS format. The procedure below describes the steps needed to create a Java JKS keystore that can be imported into ServiceNow and used for SSL client authentication.

To follow this procedure you will need to have a working JDK installed on your desktop system. This procedure was testing using OpenJDK version 1.8.0_118 however any JDK including the Sun JDK version 6 or higher should work.

3.1 Create a Public/Private key pair stored in a Java JKS keystore

```
ubuntu@ubuntu:~$ keytool -genkey -alias servicenow -keypass servicenow -keyalg RSA -keysize 2048 -keystore servicenow.jks -storepass servicenow
What is your first and last name?
[Unknown]: servicenow.testcertificates.com <--- replace testcertificates.com with your domain that is approved in your ECS account
What is the name of your organizational unit?
[Unknown]: <--- optional to provide something here
What is the name of your organization?
[Unknown]: Entrust <--- enter the name of your own Organization
What is the name of your City or Locality?
[Unknown]: Ottawa
What is the name of your State or Province?
[Unknown]: Ontario
What is the two-letter country code for this unit?
[Unknown]: CA
Is CN=servicenow.testcertificates.com, OU=Unknown, O=Entrust, L=Ottawa, ST=Ontario, C=CA correct?
[no]: yes
```
3.2 Generate a Certificate Signing Request

```
ubuntu@ubuntu:~$ keytool -certreq -alias servicenow -keypass servicenow -keystore serv|cenow.jks -storepass serv|cenow

-----BEGIN NEW CERTIFICATE REQUEST-----
encoded CSR data in Base64 format
-----END NEW CERTIFICATE REQUEST-----
```

Copy all of the output from key tool including the "BEGIN NEW CERTIFICATE REQUEST" and "END NEW CERTIFICATE REQUEST" lines into a text file. This text will be submitted to the ECS portal to request a client SSL certificate.

3.3 Create the client certificate

3.3.1 Login to the ECS portal and create a new Standard or Advantage SSL certificate.
3.3.2 Fill in the following fields on the Certificate Details tab:

**Certificate Expiry:** Select the desired expiry date. **Note:** You will need to renew this certificate and update your ServiceNow client SSL profile before this date.

**Organization:** Select an organization from the drop down list.

**Extended Key Usage:** Select “Client Authentication”.

**Certificate Signing Request (CSR):** Paste the contents of the Certificate Signing Request from keytool captured above.

3.3.3 Click on “Next”.

3.3.4 Enter any value desired such as ”ServiceNow for the “Tracking info” field.

3.3.5 Continue to click “Next” to accept each screen until prompted to confirm certificate generation.
3.3.6 Click on “Yes” to create the client SSL certificate. Your new certificate should appear at the top of the list of ECS Certificates. Record the Tracking ID of this certificate for use in the next section.

3.4 Create a new API user for Basic Authentication

3.4.1 From the menu options within the ECS portal, select “Administration > User Management”.

3.4.2 In the “Create New User” tab, enter a User Name for ServiceNow connection.

3.4.3 From the “User Type” field selection list, choose “API User”, then click on “Add this user”. The “Add New API User” dialog appears.
3.4.4 Enter the Tracking ID of the SSL certificate created in the previous section under “Client Certificate Tracking ID”. Click on “Verify” to confirm the certificate is valid for use with your API User credential.

3.4.5 Ensure that the “API Role” is set to “Super” and “Allow Auto-Approve” is checked; then click on “Add New API User”. You should receive a dialog confirming the API user was added successfully.

3.4.6 Record the Username and Password of the New API User that you have created and click “Done”.

![Add New API User](image)

**Note:** You may now copy the username and password.
3.5 Download the certificate files

3.5.1 From within the ECS portal, click on “Certificates > Managed Certificates”.

3.5.2 Select the “ECS Certificates” tab and locate your client SSL certificate created above in the list.

3.5.3 Select the certificate and select the ‘Pickup’ certificate action from the Actions drop-down menu.
3.5.4 Choose "Sun Java Web Server" as the Server type and click “Next”.

3.5.5 Click on “Download Certificates” to download the CertificateBundle.p7b file.

3.6 Import the certificate and chain into Java keystore

3.6.1 `ubuntu@ubuntu:~$ keytool -import -trustcacerts -alias servicenow -file CertificateBundle.p7b -keystore servicenow.jks`
3.7 Import the keystore into your ServiceNow instance

3.7.1 In the ServiceNow navigation bar, expand "System Security" ◊ "Protocol Profiles" ◊ click "New" on the top of the content pane.

3.7.2 In the "New" profile content pane, enter the protocol name for the SSL/TLS mutual authentication ("ecshttps" for example). Enter the SSL/TLS port. Click the "Search" button to the right of the "Keystore" box.

3.7.3 This will open a new window for selecting or creating a new key store. Click on "New" to create a new keystore.
3.7.4 Complete the form by filling in the following fields:

- **Name**: Enter a name for the certificate keystore.
- **Type**: Select Java Key Store from the drop down list.
- **Attachment**: On the top right of the screen, click the “Manage Attachments” icon. Browse to the location of the servicenow.jks keystore file created above and attach it to this record.
- **Key store password**: Enter the password for the servicenow.jks keystore.
- **Short description**: Provide a description for the ECS keystore.

When you have completed filling in the fields above, click on “Submit”.

3.7.5 Now that you have configured the keystore, click the "Submit" button on the Protocol Profile screen to complete profile creation.
3.7.6 Your new Protocol Profile should appear in the list.
4. Configure ECS outbound SOAP service

4.1 Create a new SOAP Message

4.1.1 Navigate to “System Web Service” ◊ “Outbound” ◊ “SOAP Message” in the navigation bar, then click “New” button in the content pane.

4.1.2 In the “SOAP Message” content pane, enter the following information:
- **Name**: Enter a name for the ECS Web Service API.
- **WSDL**: Click the lock icon to edit the WSDL and enter the URL for the ECS service endpoint.
- **Download WSDL**: Ensure the box to download the WSDL is checked.
- **Use mutual authentication**: Ensure the box to use mutual authentication is checked.
- **Protocol profile**: Select the Protocol Profile for client authentication created in the steps above.

When you have completed filling in these fields, click “Submit”. 
4.1.3 Your new SOAP service should appear in the list.

4.2 Test the Service

4.2.1 To test that the service has been configured properly, start by clicking the Name of the service (e.g. ECSWebServiceAPI) in the SOAP Messages list.

4.2.2 In the “Soap Message” pane, click on the link “Generate sample SOAP messages” toward the bottom of the screen.
4.2.3 The entire WSDL should be populated into the "WSDL XML" field.

Also, at the bottom of the page, all functions in defined in the WSDL should be listed.

4.2.4 SOAP calls to the ECS API requires Basic Authentication. To configure this, click on one of the functions in the list. The following example below uses getDomainList.
4.2.5 Set the "Authentication type" to "Basic", then click the "Search" icon to the right of the "Basic auth profile" field.

4.2.6 In the “Basic Auth Configurations” window, click on “New” to create a profile for Basic Auth.

4.2.7 Enter a name for the ECS API profile, then enter the Username and Password for the API User created from within the ECS portal in the steps above. After filling in these fields, click "Submit". **Note**: The same Basic Auth profile can be used to configure any other ECS API function.
When you have returned to the “SOAP Message Function” window, complete the configuration as follows:

**Authentication type**: Basic

**Basic auth profile**: Set to profile created in the previous step (e.g. ECS API User).

**Use mutual authentication**: Ensure this box is checked.

**Protocol profile**: Set to profile containing your client certificate (e.g. ecshttps).

Once these fields are filled in, click “Update” at the bottom of the window.

From the “SOAP Message” window, scroll back down to the list of SOAP Message Functions and click on the function you were just configuring (e.g. `getDomainList`).
4.2.10 Scroll down to the “Variable Substitutions” section near the bottom of the screen. Click on “New” to add testing API arguments.

![Variable Substitutions](image)

4.2.11 In the “Variable Substitutions” form, enter the name and value of the variable and click “Submit”. (For `getDomainList`, set test value `clientID=1`)

![Variable Substitutions Form](image)

4.2.12 Click on "Test" in the "Related Links" section to launch the API test.

**Related Links**
- Auto-generate variables
- Preview Script Usage
- Refresh SOAP message
- Test
4.2.13 If everything is configured properly, the response to the SOAP call should appear in the Response field.

4.3 Scripting calls to the ECS API

At this point, ServiceNow has been configured to make calls to the ECS API. For integration, a developer will need to know what code (javascript) is invoked to perform an API function call.

To review the code, from within the “SOAP Message Function” screen, click on “Preview Script Usage” in the “Related links” section.
This will open a new browser window which contains the sample script for the specific API. For example, here is the code for the function `getDomainList`:

```javascript
try {
    var s = new sn_ws.SOAPMessageV2('x_85144_entrust_ce.ECSWebServiceAPI', 'getDomainList');

    // override authentication profile
    // authentication type = 'basic'
    // r.setAuthentication(authentication type, profile name);

    s.setStringParameter('clientID', '1');
    var response = s.execute();
    var responseBody = response.getBody();
    var status = response.getStatusCode();
}
catch (ex) {
    var message = ex.getMessage();
}
```

This code can be used in ServiceNow Business Rules, UI Actions, or Workflow, depending on the use case. Notice that this script only stores the SOAP response as a raw XML string. Developers will need to add additional code to parse the SOAP response. This can be done by using ServiceNow APIs. For example:

```javascript
var xmlDoc = new XMLDocument2();
xmlDoc.parseXML(responseBody);
var errorId = xmlDoc.getNode('//certRequestReturn[1]').textContent().trim();
```

For ServiceNow API usage, please refer to the ServiceNow API document.
5. Create certificate request application

This section is a step by step guide of creating an ad-hoc application to request/approve a certificate from ECS. Developers may find this is helpful if starting from scratch.

Once the application development is completed, a developer should export the entire application to an XML file. The system integrator should be able to import and install the application to another ServiceNow instance.

It is assumed we have an empty custom application (e.g. Entrust Certificate Services) created and we have completed configuring the outbound ECS soap service.

5.1 Create a database table to store the CSR and certificate from ECS

5.1.1 To access the existing application, in the navigation bar, select "System Applications" > Applications.

5.1.2 In the content pane, click the “Edit” button to the right of the application to launch the ServiceNow Studio.
5.1.3 In ServiceNow Studio, click "Create New Application File" button on the top left corner.

5.1.4 In the "Create New Application File" window, select "Data Model" > "Table" > "Create".
5.1.5 In the “New Table” content pane, enter a Label name (e.g. certRequest) and click "Submit". This will create a new table with some prepopulated columns.

5.1.6 Add a column to store the Certificate Signing Request. In the newly created table, in the "Columns” tab, click “New” button to create new column.
5.1.7 In the "Dictionary Entry" window, fill in the fields as below:

- **Type:** String
- **Column label:** Certificate Signing Request
- **Column name:** csr
- **Max length:** 4000

5.1.8 After filling in the fields, click on “Submit” to add the column. The new column will appear in the table as shown below.
5.1.9 Repeat the same process to add table columns for the following values.

<table>
<thead>
<tr>
<th>Label</th>
<th>Name</th>
<th>Type</th>
<th>Max length</th>
<th>Mandatory</th>
<th>Default Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Additional Emails</td>
<td>additionalEmails</td>
<td>String</td>
<td>100</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Auto Approve</td>
<td>autoapprove</td>
<td>True/False</td>
<td>40</td>
<td>True</td>
<td></td>
</tr>
<tr>
<td>Cert Expiry</td>
<td>certexpdate</td>
<td>Date/Time</td>
<td>40</td>
<td>Checked</td>
<td></td>
</tr>
<tr>
<td>Certificate Type</td>
<td>certtype</td>
<td>String</td>
<td>40</td>
<td>Checked</td>
<td>Standard</td>
</tr>
<tr>
<td>Client</td>
<td>clientid</td>
<td>Integer</td>
<td>40</td>
<td>Checked</td>
<td>1</td>
</tr>
<tr>
<td>Extended Key Usage</td>
<td>eku</td>
<td>String</td>
<td>40</td>
<td></td>
<td>ServerAndClientAuth</td>
</tr>
<tr>
<td>Organizational Unit</td>
<td>ou</td>
<td>String</td>
<td>40</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Request Type</td>
<td>requesttype</td>
<td>String</td>
<td>40</td>
<td>Checked</td>
<td>New</td>
</tr>
<tr>
<td>Requester Email</td>
<td>appemail</td>
<td>String</td>
<td>40</td>
<td>Checked</td>
<td></td>
</tr>
<tr>
<td>Requester Name</td>
<td>appname</td>
<td>String</td>
<td>40</td>
<td>Checked</td>
<td></td>
</tr>
<tr>
<td>Requester Phone</td>
<td>apptelephone</td>
<td>String</td>
<td>40</td>
<td>Checked</td>
<td></td>
</tr>
<tr>
<td>SANs</td>
<td>subjectAltName</td>
<td>String</td>
<td>4000</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Tracking ID</td>
<td>trackingid</td>
<td>String</td>
<td>40</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

5.2 Create a Form to collect user input

5.2.1 Edit your application to launch ServiceNow Studio.

5.2.2 In ServiceNow Studio, click “Create New Application File” button on the top left corner.
5.2.3 In the "Create New Application File" window, select "Forms & UI" and click "Next".

5.2.4 In the "Create New Application File" window, make sure the "certRequest" table is highlighted and click "Create".
5.2.5 In the “Form Design” window, rearrange the form fields by dragging and dropping them from one section to another. You can also rename sections and fields by clicking on the name.

5.2.6 Move the mouse to the “Request Type” field, and click “Edit this field” icon.
5.2.7 Make sure this field is marked "Mandatory", and close the properties editor.

5.2.8 Repeat the same process to confirm the following fields are also marked "Mandatory":
- Certificate Type
- Client
- Certificate Expiry
- Requester Name
- Requester Email
- Requester Phone
5.2.9 Edit the “Request Type” field again and scroll down to the bottom of the “Properties” window to the section labelled “Choices”. Set the “Choice type to “Dropdown with none”. Add the choices “New”, “Reissue” and “Renew” to the list.

5.2.10 Repeat the same process to create the following choice values for the “Certificate Type” field:
- Advantage [advantage]
- EV Multi-Domain [ev]
- Standard [standard]
- UC Multi-Domain [ucc]
- Wildcard [wc]
5.2.11 Repeat the process again to create the following choice values for the "Client" field:
- Primary [1]
- Client 2 [2]
- Client 3 [3]

5.2.12 Once you are satisfied with the Form layout, click “Save” in the Form Design window.
5.2.13 At this point, the Form Design has been completed. To test the form, search for your application (e.g. "Entrust Certificate Services") in the navigation menu and select the database table “certRequests”.

5.2.14 A list view of the database table is displayed. Click the “New” button on the top of the screen.

5.2.15 The screen changes to the “certRequests” form view. The end user would use this form to submit certificate requests. Review this form to ensure you are satisfied with your design.
5.3 Add a Business Rule to invoke the ECS API to submit the CSR

In this step we will fill out the “certRequests” form and send the certificate request into ECS. To accomplish this, we need to add a Business Rule to trigger the ECS API service call.

5.3.1 Navigate to the “System applications” > ”Application” and click the “Edit” button next to the application to launch ServiceNow Studio.

5.3.2 In ServiceNow Studio, click “Create New Application File” button on the top left corner.
5.3.3 In the "Create New Application File" window, select "Server Development" > "Business Rule" > "Create".

5.3.4 In the "New Business Rule" tab, enter the "Name", select the table (e.g. "certRequests"), and check the "Advanced" check box. In the "When to run" tab, check the "Insert" check box.
5.3.5 In this step, we associate a script to the “Submit” button on the “New certRequest” form. After the user fills out the form and clicks the “Submit” button, the certificate request will be sent to the ECS API. Click the “Advanced” tab. Copy the code in Appendix A and paste it into the “Script” editor.

5.3.6 Click the “Submit” button to complete the Business Rule.
5.4 Add a UI Action to approve the pending certificate

By default, the ECS API will automatically approve certificate requests (autoapprove=True) and respond by sending back the certificate. Alternatively, the ServiceNow application developer could choose to have certificate requests queued for administrative approval (autoapprove=False), in which case ECS would respond with the Tracking ID of the pending request. The response to certificate requests submitted to ECS is added to the “certRequest” table list view.

In the case where there are pending certificate requests that need to be approved, a "UI Action" can be added to initiate the approval process.

5.4.1 In ServiceNow Studio, click "Create New Application File".
In the "Create New Application File" window, select "Server Development" > "UI Action" > "Create".
5.4.3 In the "New UI Action" window, fill out the fields as follows:

**Name**: Enter a name for the UI Action.

**Table**: Select the table (e.g., "certRequest") from the list.

**List context menu**: Ensure this box is checked.

**Condition**: In this field, enter `current.status.getDisplayValue() == 'Pending'`

**Script**: In the script editor, cut/paste the code from Appendix B.

5.4.4 Click the "Submit" button to complete the UI Action.
5.4.5 Now for any records in the database table list view, right clicking the mouse will display a context menu containing an option for the newly added “UI Action”. Selecting that option will trigger the corresponding script to run, which will approve the pending certificate request (if it is in Pending state) by sending a request to the ECS API.
6. Publish the application

Once the Entrust Certificate Services application is complete, you may want to publish it.

6.1 Navigate to “System Application” > “Applications” and click the icon next to the application.

6.2 In the “Custom Application” form, select “Publish to Update Set” in the “Related links” section.
6.3 In the “Publish to Update Set” page, click “Publish”.

![Publish to Update Set window]

6.4 Once the process has completed, click “Done” to continue.

![Progress window]
6.5 This will bring up the "Update Set" form. Click the "Export to XML" link in the "Related Link" section.

6.6 The entire application data will be saved in a XML file. This XML file can be imported to another ServiceNow instance.
7. Appendix A: Sample javascript for CSR submission

(function executeRule(current, previous /*null when async*/) { 
try {
    var s = new sn_ws.SOAPMessageV2('x_85144_entrust_ce.ECSWebServiceAPI', 'certRequest');
    var aap = 'true';
    // Populate the soap stub function arguments from the current database record
    s.setStringParameter('certtype', current.certtype);
    s.setStringParameter('clientid', current.clientid);
    s.setStringParameter('requesttype', current.requesttype);
    s.setStringParameter('appemail', current.appemail);
    s.setStringParameter('apptelephone', current.apptelephone);
    s.setStringParameter('csr', current.csr);
    s.setStringParameter('appname', current.appname);
    s.setStringParameter('certexpdate', current.certexpdate.getDisplayValue().split(' ').join('T'));
    // Have to call 'executeAsync'. It's not clear at the moment why 'execute' does
    // not work.
    var response = s.executeAsync();
    // Parsing the response XML
    var xmlDoc = new XMLDocument2();
    xmlDoc.parseXML(responseBody);
    var errorId = xmlDoc.getNode('//certRequestReturn[1]').getTextContent().trim();
    var errorTitle = xmlDoc.getNode('//certRequestReturn[2]').getTextContent().trim();
    var errorText = xmlDoc.getNode('//certRequestReturn[3]').getTextContent().trim();
    if(errorTitle == null || errorTitle == '')
        errorTitle = null;
    if(errorText == null || errorText == '')
        errorText = null;
    // Received error from ECS
    if(errorTitle || errorText ){
        throw "Error ": errorTitle + ":" + errorText;
    }
    var serialNumber = xmlDoc.getNode('//certRequestReturn[4]');
    current.serialnumber = serialNumber.getTextContent();
    var serverCert = xmlDoc.getNode('//certRequestReturn[5]');
    current.servercert = serverCert.getTextContent();
    var chainCert = xmlDoc.getNode('//certRequestReturn[6]');
    current.chaincert = chainCert.getTextContent();
    var certExpiryDate = xmlDoc.getNode('//certRequestReturn[7]');
    current.certexpirydate = certExpiryDate.getTextContent();
    var trackingId = xmlDoc.getNode('//certRequestReturn[8]');
    current.trackingid = trackingId.getTextContent();
    var pickupUrl = xmlDoc.getNode('//certRequestReturn[9]');
    current.pickupurl = pickupUrl.getTextContent();
    if(aap == 'true'){
        var smimePkcs12 = xmlDoc.getNode('//certRequestReturn[10]');
        current.smimepkcs12 = smimePkcs12.getTextContent();
        current.status = 'Active';
    }else{
        current.status = 'Pending';
    }
})
try {
    current.setAbortAction(false);
    return true;
} catch(ex) {
    gs.error("Soap Service error: {0}", ex);
    // Abort the operation.
    current.setAbortAction(true);
}

})(current, previous);
try {
    var s = new sn_ws.SOAPMessageV2('x_85144_entrust_ce.ECSWebServiceAPI', 'certApprove');
    s.setStringParameter('trackingid', current.trackingid);
    s.setStringParameter('appname', current.appname);
    s.setStringParameter('apptelephone', current.apptelephone);
    s.setStringParameter('appemail', current.appemail);
    var response = s.execute();
    var responseBody = response.getBody(); 
    var status = response.getStatusCode();
    // Parsing soap response.
    var xmlDoc = new XMLDocument2();
    gs.info("Parsing response");
    xmlDoc.parseXML(responseBody);
    var errorId = xmlDoc.getNode('//certApproveReturn[1]').getTextContent().trim();
    var errorTitle = xmlDoc.getNode('//certApproveReturn[2]').getTextContent().trim();
    var errorText = xmlDoc.getNode('//certApproveReturn[3]').getTextContent().trim();
    if(errorTitle == null || errorTitle == '')
        errorTitle = null;
    if(errorText == null || errorText == '')
        errorText = null;
    // Received error from ECS
    if(errorTitle || errorText ){
        throw "Error "+ errorTitle + ":" + errorText;
    }
    // Update database table
    var serialNumber = xmlDoc.getNode('//certApproveReturn[4]');
    current.serialnumber = serialNumber.getTextContent();
    var serverCert = xmlDoc.getNode('//certApproveReturn[5]');
    current.servercert = serverCert.getTextContent();
    var chainCert = xmlDoc.getNode('//certApproveReturn[6]');
    current.chaincert = chainCert.getTextContent();
    var certExpiryDate = xmlDoc.getNode('//certApproveReturn[7]');
    current.certexpirydate = certExpiryDate.getTextContent();
    var trackingId = xmlDoc.getNode('//certApproveReturn[8]');
    current.trackingid = trackingId.getTextContent();
    var pickupUrl = xmlDoc.getNode('//certApproveReturn[9]');
    current.pickupurl = pickupUrl.getTextContent();
    current.status = "Active";
    current.update();
}catch(ex) {
    var message = ex.getMessage();
    gs.error("Soap Service error: {0}", ex);
}
Consumers, citizens and employees increasingly expect anywhere-anytime experiences — whether they are making purchases, crossing borders, accessing e-gov services or logging onto corporate networks. Entrust Datacard offers the trusted identity and secure transaction technologies that make those experiences reliable and secure. Solutions range from the physical world of financial cards, passports and ID cards to the digital realm of authentication, certificates and secure communications. With more than 2,000 Entrust Datacard colleagues around the world, and a network of strong global partners, the company serves customers in 150 countries worldwide.

For more information about Entrust products and services, call 888-690-2424, email entrust@entrust.com or visit www.entrust.com.

Entrust Datacard and Entrust are trademarks, registered trademarks and/or service marks of Entrust Datacard Corporation in the United States and/or other countries. Names and logos on sample cards are fictitious. Any similarity to actual names, trademarks or tradenames is coincidental.

©2016 Entrust Datacard Corporation.
All rights reserved.

Headquarters
Entrust Datacard
1187 Park Place
Shakopee, MN 55379
USA