Installing and configuring ETSI MSS Mobile PKI - SSO

The list below provides an overview of the main phases in the installation process.

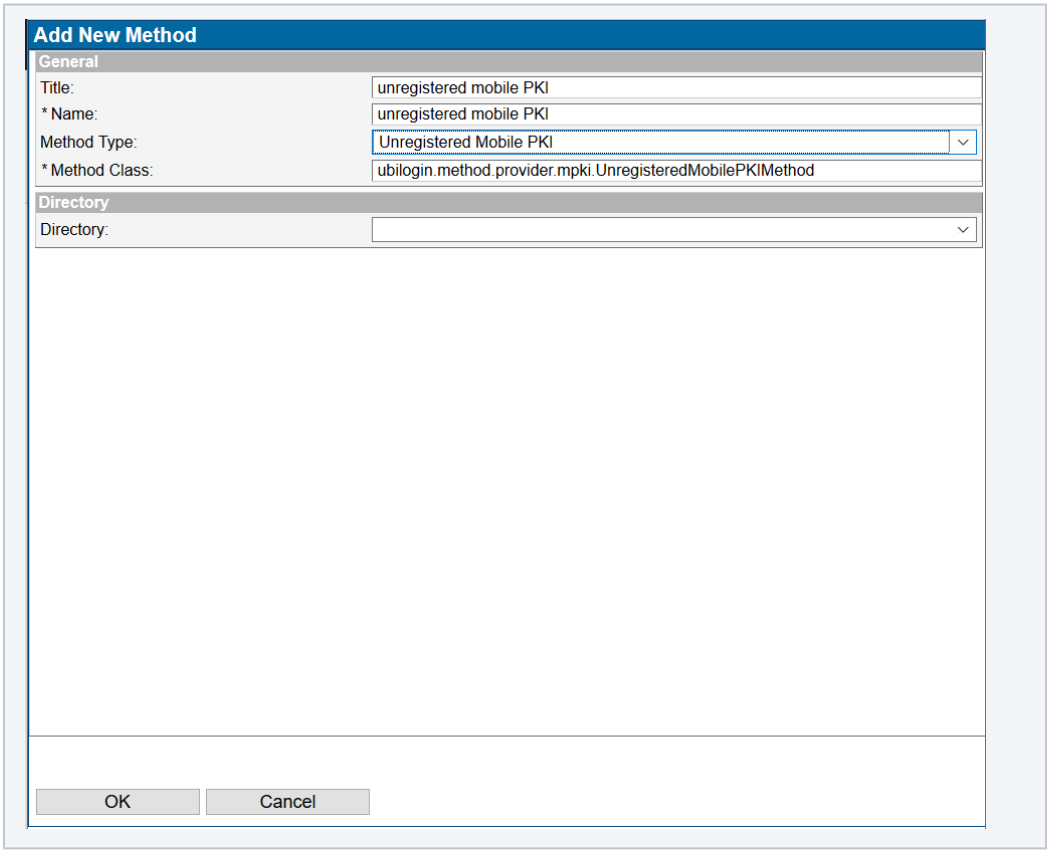
1. Decide whether to create an unregistered or registered MPKI method
2. Decide which personal identity attribute (if any) SSO Server should query from the MSSP regarding the user.
3. Create the method in SSO management application
4. Create the following files: etsimss.properties, policy.xml, ssl-policy.xml
5. Edit config.index to attach the etsimss.properties file to the method
6. Test the configuration

The following sections provide detailed information regarding the phases:

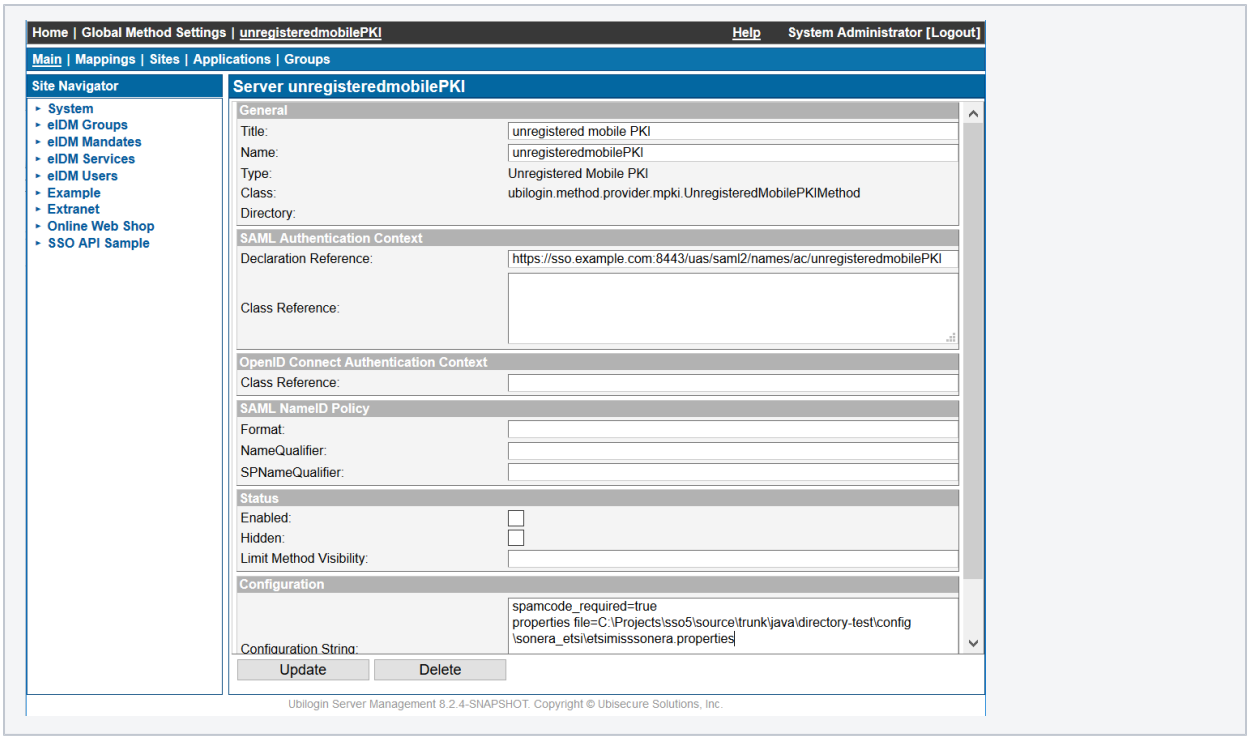
Creating the Mobile PKI Method

To create a new authentication method in Ubisecure SSO Management application:

1. Select "**Global Method Settings**" **"New Method…"**
2. Give a name for the method and select 'Unregistered Mobile PKI' or 'SPI Mobile PKI' as the method type. If 'SPI Mobile PKI' is chosen a directory must be specified.



3. Click **OK**
4. Enable the method by checking the "Enabled" checkbox as shown in the following figure:



Note the configuration parameters:

- properties.file points to etsimss.properties file
- spamcode_required put "false" if you want the users to try the login first without the spamcode

Configuring the Mobile PKI Method

The next step is to configure the method. This is done in *etsimss.properties* file. The properties file holds all the configuration information and references to the certificates. These certificates and configuration strings are determined from the resources the MSSP has provided.
Create a new folder *methods/etsimss* in the SSO installation directory. After the steps defined in this chapter, the folder should contain the following files: *etsimss.properties*, *client.p12*, *ssl.xml*, *policy.xml*

Required Certificates and XML Files for Storing Them

The following table describes the required certificates and the XML files for storing them:

File	Content	Description
client.p12	The client certificate in PKCS12 format with private key included in the file.	The client certificate used to identify the Ubisecure SSO Server to the MSSP for the SSL connection.

<table>
<tr>
<td>ssl-policy.xml</td>
<td>To create the proper ssl.xml file:

1. Export the CA certificate in base64 format.
2. Insert the base64 format certificate to the xml-template inside the <Trust> tags as shown below in the example:

<pre><?xml version="1.0" encoding="iso-8859-1"?>
<Policy xmlns="http://ubisecure.com/schema/certagent.xsd">

<PKI>
<Trust>
MIIDYDCCAkigAwIBAgIQunHfEI8HO5lDaGQF9jV/bzANBgkqhkiG9w0BAQUFADBS
MQswCQYDVQQGEwJGSTEgMB4GA1UEChMXVWJpc2VjdXJlIFNvbHV0aW9ucyBJbmM
PprFT4FGEN53lIBuB44NURE1W+XWWA/2rJYJcLBz49gyLgxo32C1dgvoTrrpdjve
buC6jm9QtXlPrxIQ7N2IaHCUf4SLi7NxZGAn28fYA4dKgpiM3W8HS1xjH7IELxps
0QkseA==
</Trust>
</PKI>

</Policy></pre></td>
<td>The MSSP CA server certificate in base64 form. The CA certificate is used to identify the MSSP when creating the SSL connection.</td>
</tr>
<tr>
<td>policy.xml</td>
<td>To create a proper policy.xml file:

1. Export the certificate in base64 format.
2. Insert the base64 format certificate to the xml-template inside the <Trust> tags as shown below in the example. (The base64 certificate should be placed.)

<pre><?xml version="1.0" encoding="iso-8859-1"?>
<Policy xmlns="http://ubisecure.com/schema/certagent.xsd">

<PKI>
<Trust>
MIIDYDCCAkigAwIBAgIQunHfEI8HO5lDaGQF9jV/bzANBgkqhkiG9w0BAQUFADBS
MQswCQYDVQQGEwJGSTEgMB4GA1UEChMXVWJpc2VjdXJlIFNvbHV0aW9ucyBJbmM
PprFT4FGEN53lIBuB44NURE1W+XWWA/2rJYJcLBz49gyLgxo32C1dgvoTrrpdjve
buC6jm9QtXlPrxIQ7N2IaHCUf4SLi7NxZGAn28fYA4dKgpiM3W8HS1xjH7IELxps
0QkseA==
</Trust>
</PKI>

</Policy></pre></td>
<td>The certificate that is used to confirm the signature the MSSP provides to Ubisecure SSO Server at the end of the transaction. The concept of the policy.xml file is same as in ssl.xml.</td>
</tr>
</table>

etsimss.properties File

This file holds the information of the configuration strings and references to the three files defined in the table above. The various attributes that must be used in the configuration file are provided in the table below:

NOTE: The *policy.xml* , *ssl-policy.xml* and *client.cert* attributes define a file location while the other attributes define a configuration string.

Attribute description

```
Attribute = Value
```

apId defines the Application Provider's unique URI-type identifier with which the AP is registered to use the AE's services.

```
apId = test.server.fi/test
```

apPwd defines the password used in authenticating the Application Provider.

```
apPwd = 4f344534
```

Client.cert defines the path to the client certificate.

```
client.cert = client.p12
```

client.cert.password defines the password used for protecting the client certificate and the private key.

```
client.cert.password = fD2s&#hJ
```

ssl-policy.xml defines the path to the XML file containing the MSSP root CA in base64 form. This certificate is used to check the MSSP server certificate when creating the SSL connection.

```
ssl-policy.xml = ssl-policy.xml
```

policy.xml defines the path to the XML file containing the certificate that is used to confirm the signature the MSSP provides to Ubisecure SSO.

```
policy.xml = policy.xml
```

ae.signatureUrl defines the URL for making signature requests to the MSSP.

```
ae.signatureUrl = https://server.fi/MSS_Signature
```

ae.statusUrl defines the URL for making status requests to the MSSP.

```
ae.statusUrl = https://server.fi/MSS_Status
```

personIdentityAttributes defines the attributes that SSO Server will query from the MSSP regarding the user. Multiple personIdentityAttributes are separated with a white space character. This is the only configuration string where you have to make a decision. It is essential to define this attribute if you are using an unregistered MPKI method.
The definition of this attribute is not mandatory. If the personIdentityAttributes is not defined, no attributes will be queried from the MSSP.

```
personIdentityAttributes = http://mss.ficom.fi/TS102204/v1.0.0/PersonID#hetu
```

ae.timeout *(optional)* defines how long SSO keeps querying the MSSP before terminating the authentication process. The value is in minutes. Default value is 5.

```
ae.timeout = 5
```

ae.signatureProfile *(optional)* defines the signature profile. The only supported signatureprofile is authentication profile (http://mss.ficom.fi/TS102206/v1.0.0/authentication-profile.xml). However, this profile can be overridden with the supplied value.

```
ae.signatureProfile = http://mss.ficom.fi/TS102206/v1.0.0/authentication-profile.xml
```

xml.parser.validation *(optional)* defines whether the xml responses will be validated against the schema. **Use for debugging purposes only**.

```
xml.parser.validation = false
```

eventIdLength *(optional)* defines the length of generated EventID. Default value is 8.

```
eventIdLength = 7
```

initialStatusRequestDelay *(optional)* defines the delay before the first status request is sent after the initial transaction request. The value is in milliseconds. Default value is 15000.

```
initialStatusRequestDelay = 4000
```

consecutiveStatusRequestDelay *(optional)* defines the delay of the consecutive status requests after the first status request. The value is in milliseconds. Default value is 5000.

```
consecutiveStatusRequestDelay = 1000
```

threadPoolSize *(optional)* defines the number of threads available in the thread pool dedicated to processing ETSIMSS requests and responses. Minimum value is 1. Maximum value is 20. Default value is 1.

```
threadPoolSize = 2
```

After all the necessary Attribute values in *etsimss.properties* configuration file have been set, the file's contents should look similar to the example below:

```
apId = test.server.fi/rajapinta-xxxxxxxxx
apPwd = app_password
#cert = org
client.cert = client cert.p12
client.cert.password = password


ae.signatureUrl = https://localhost:444/MSS_Signature
ae.statusUrl = https://localhost:444/MSS_StatusPort
ae.receiptUrl = https://localhost:444/MSS_ReceiptPort


ssl-policy.xml = sslpolicy.xml
ae.timeout = 90
ae.msspId.dnsName = mssp2.localgost
policy.xml = policyxml.xml


personIdentityAttributes = http://mss.ficom.fi/TS102204/v1.0.0/PersonID#hetu http://mss.ficom.fi/TS102204/v1.0.0
/PersonID#satu
http://mss.ficom.fi/TS102204/v1.0.0/PersonID#age http://mss.ficom.fi/TS102204/v1.0.0/PersonID#ageClass
http://mss.ficom.fi/TS102204/v1.0.0/PersonID#gender http://mss.ficom.fi/TS102204/v1.0.0/PersonID#givenName
http://mss.ficom.fi/TS102204/v1.0.0/PersonID#surName http://mss.ficom.fi/TS102204/v1.0.0/PersonID#subject
http://mss.ficom.fi/TS102204/v1.0.0/PersonID#validUntil

eventIdLength = 7
initialStatusRequestDelay = 4000
consecutiveStatusRequestDelay = 1000
threadPoolSize = 2
```

Correct methods/etsimss Directory Contents

The *methods/etsimss* directory should contain at least the files shown in Figure 3 with the contents described above.

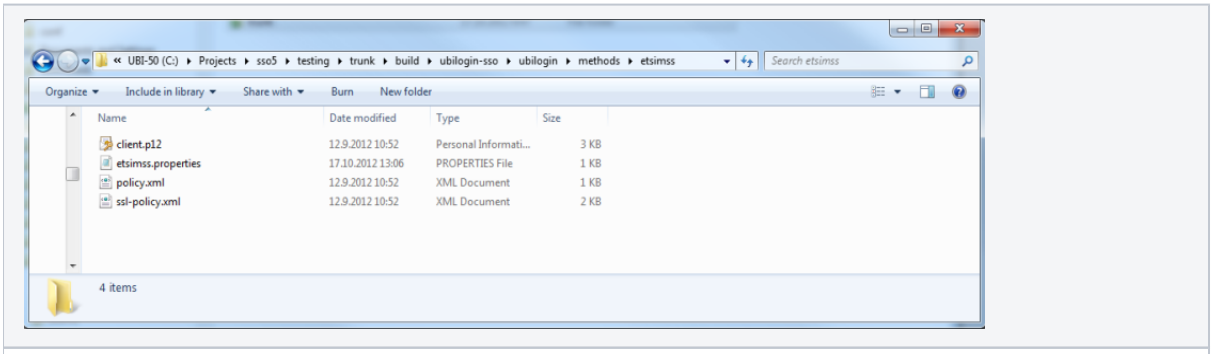


Figure 3 methods/etsimss directory with the required files

Configuring An Application to Use the Mobile PKI Method

This chapter describes how to configure an application to use the mobile PKI method.

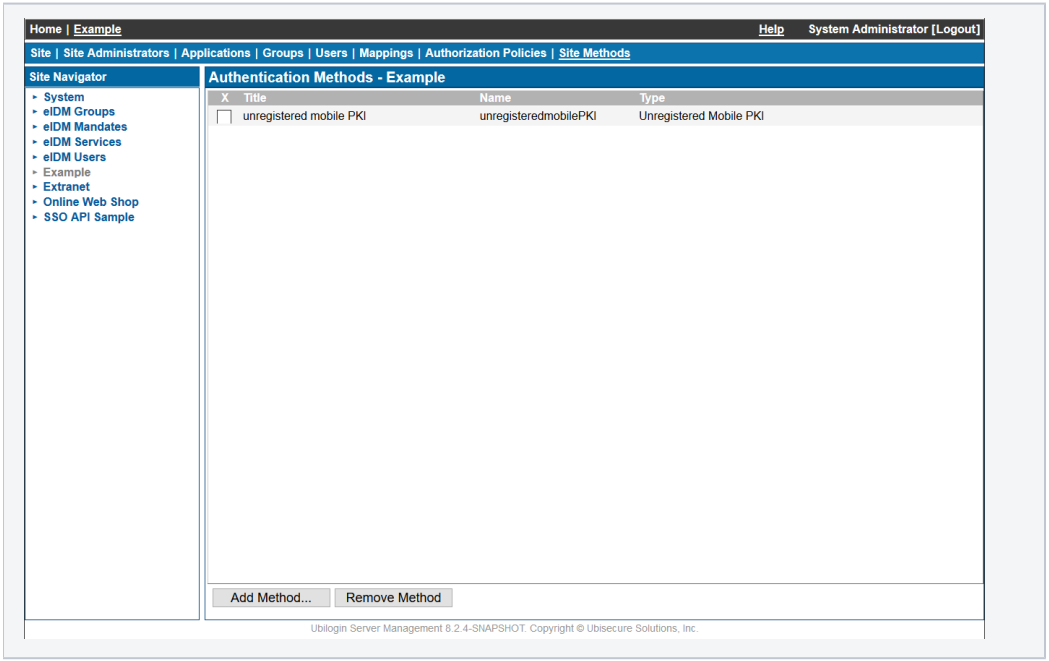
Prerequisites

- A mobile PKI method has been created and configured as explained in the previous sections.
- An appropriate type of application has been integrated to Ubisecure SSO. For instance, "SAML SP for Java.

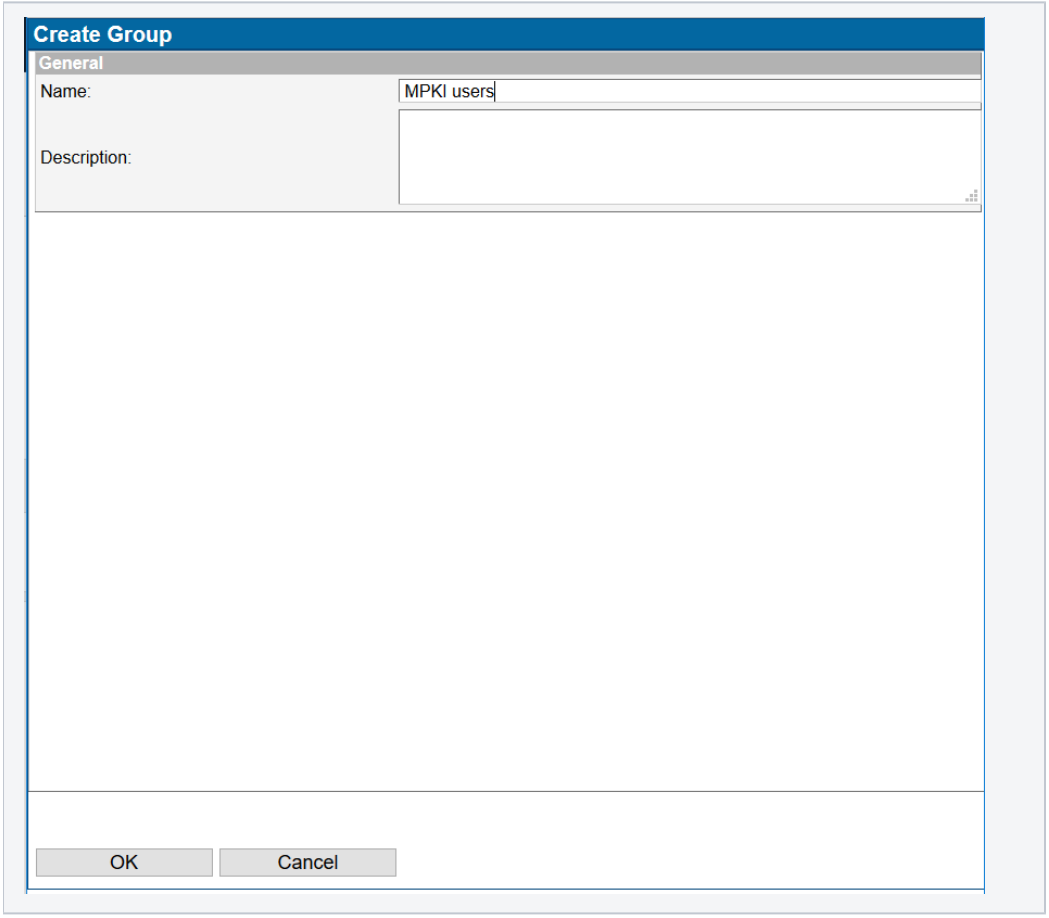
> ⚠ **NOTE:** the "Sample application" available in Ubisecure extranet can be used to verify the configuration.

Application Configuration Process for Unregistered Mobile PKI method

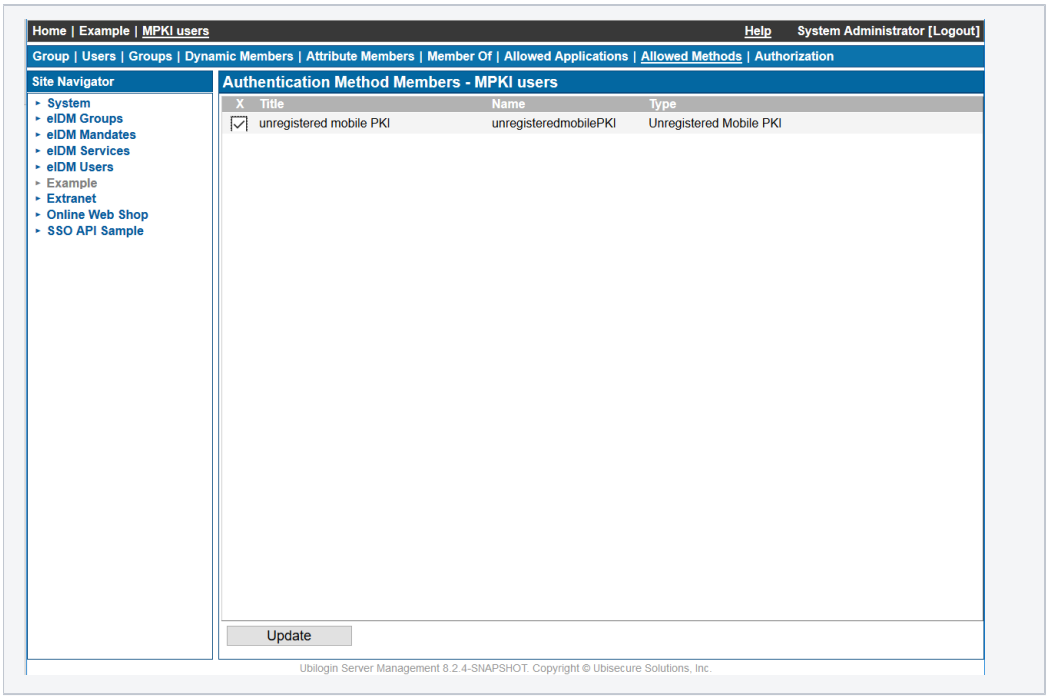
Select the site where the application is to be created and add the mpki.etsi.1 to the methods of the site



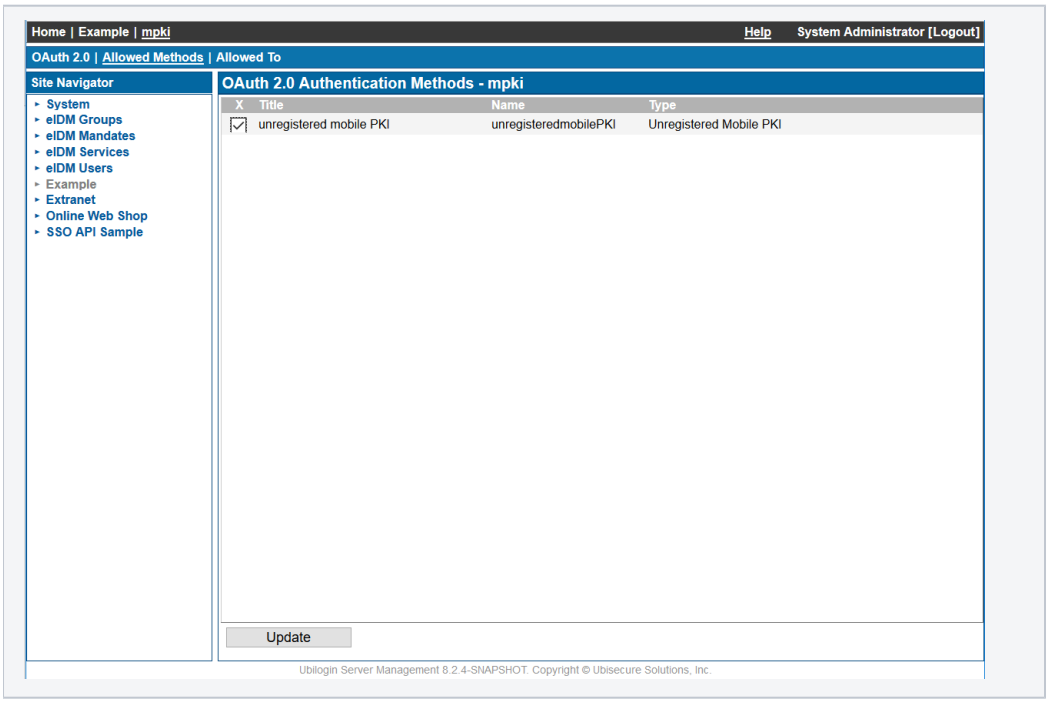
Create a group in the site. Select **"Groups" "New Group"**



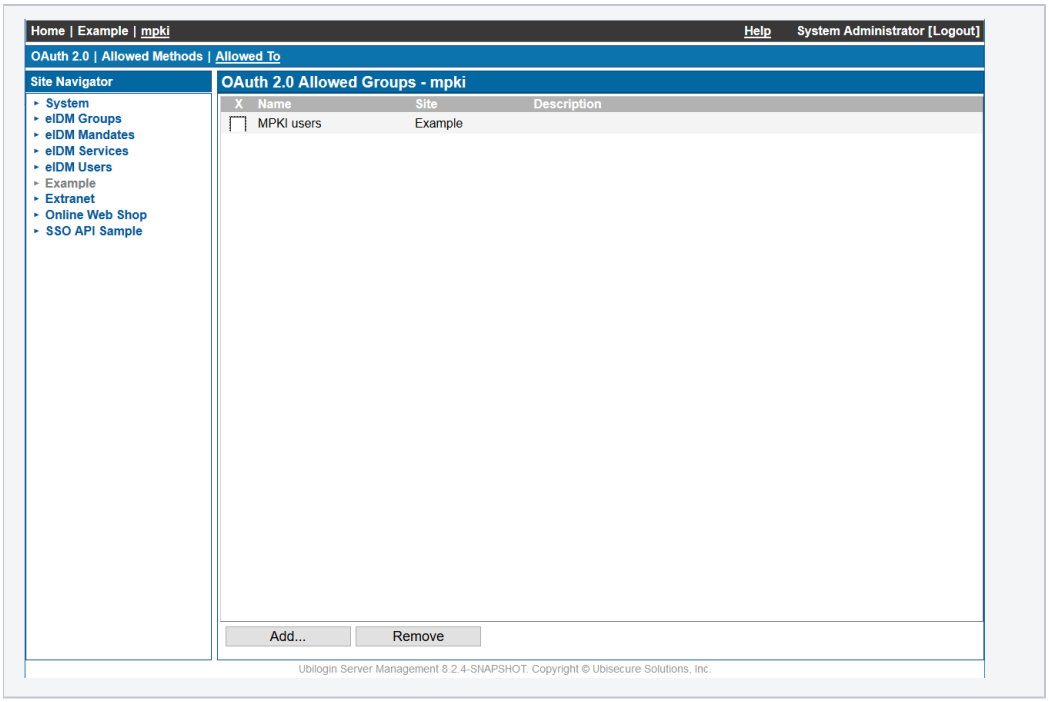
Enable mpki.etsi.1 method in the group



Enable mpki.etsi.1 method in the application



Add the mobile PKI users group to the "Allowed To" list in the application



Application Configuration Process for Registered Mobile PKI method

The application configuration steps for the registered Mobile PKI method are the same as above with two exceptions:

- The group must contain the users
- The method type must be "SPI Mobile PKI" (selected in Mobile PKI method creation)