Hard Token Management Framework

ToLiMa, Installation and Configuration Manual

Written By
Philip Vendil, philip@primekey.se
+46709885814

2007-07-29

1 Introduction/Scope

This document describes the steps involved in installing and configuring the ToLiMa application or other custom developed applications using the Hard Token Management Framework (HTMF).

The ToLiMa application have been designed to be easy to use for token administrators and therefore well integrated with the organization's other systems. This makes it a bit complicated to set up since there is a need for custom developed plug-ins to receive user data and to publish generated certificates to authorization systems in order to use ToLiMa with full functionality.

For more technical details about the framework there is a developers handbook at www.hardtokenmgmt.org.

The Hard Token Management Framework is an add-on the EJBCA (www.ejbca.org) and is developed as an Java-applet and accessed through a browser.

1.1 Who should read this document.

The intended audience of this document is system administrators interested in setting up ToLiMa or other custom HTMF projects in their organizations. It is preferred that the reader has some knowledge about PKI and smart cards but don't need be an expert in any of the areas.

2 Document History

Version	*Date*	*Name*	*Comment*
1.0	2007-07-29	Philip Vendil	First official version of the document

Table of Contents

1 Introduction/Scope....2
1.1 Who should read this document....2
2 Document History....2
3 Preparations....4
3.1 What is needed....4
3.2 Hardware and PKCS11....4
3.3 Applet jar signing key store....4
3.4 Customized User Data Source (Optional)....5
3.5 Customized Publisher (Optional)....5
4 Installing and configuring the server....6
4.1 Configuring HTMF....6
4.1.1 The hardtokenmgmt.properties....6
4.1.2 Configuring the Global Settings....7
4.2 Deploying HMTF to EJBCA....9
4.3 Configuring EJBCA....10
4.3.1 During build time....10
4.3.2 In the Web GUI....10
4.3.2.1 Create the Activation Publisher (Optional)....10
4.3.2.2 Create the User Data Source (Optional)....10
4.3.2.3 Create the CAs used by ToLiMa....10
4.3.2.4 Create the Certificate Profiles....11
4.3.2.5 Create the End Entity Profile....11
4.3.2.6 Administrative Privileges....11
4.3.2.7 System Configuration....12
4.3.3 Import Hard Token Data....13
4.4 Finally....14
5 Installing and configuring the client workstation....15
5.1 Required software and hardware....15
5.2 Generating the client installer....15
5.3 The first administration card....15
6 Starting up the application....15
7 More Information....16

3 Preparations

This chapter describes the initial preparations needed to be done before the installation process can start.

3.1 What is needed

General HTMF Application

- The Hard Token Management Framework (www.hardtokenmgmt.org)
- One EJBCA installation, version 3.5 and later. (www.ejbca.org)
- The Ant build tool (Should have already been installed with EJBCA)
- 1 Java key store (JKS) used to sign the jar files. (see below)
- Two card readers for each token management workstation.
- At least two of the supported tokens.
- A compatible PKCS11

ToLiMa application

- One customized user data source (Optional)
- One customized publisher (Optional)

3.2 Hardware and PKCS11

The tokens you are going to issue must be supported by the HTMF, either through own development or by using one of the supported hardware listed at http://www.hardtokenmgmt.org/supportedtokens.html.

In order to get going you need at least two cards, one for the token administrator to authenticate himself and the other to process using the application.

HTMF uses the standard interface PKCS11 (usually but it isn't required) to communicate with the card and can be seen as drivers for the card. This PKCS11 must have been tested with the HTMF before it is used.

To see a list of tested PKCS11 also see http://www.hardtokenmgmt.org/supportedtokens.html.

Regarding card readers should most readers supported by the PKCS11 work but HTMF have mostly been tested with the OmniKey 3210 which have proved to be a very robust.

3.3 Applet jar signing key store

In order for the HTMF applet to access local resources on the computer it must be signed by a key store that the workstation trusts. Therefore you have to create a Java key store signed by your trusted PKI. For testing and development purposes is HTMF shipped with a test key store called jarsigner.jks.

3.4 Customized User Data Source (Optional)

ToLiMa supports and is recommended to be used with an EJBCA User Data Source plug-in in order to automatically fetch user data given a unique user id such as personal number or employee number.

See EJBCA documentation at http://www.ejbca.org/manual.html#Framework%20for%20External%20User%20Data%20Sources for instructions how to write such a plug-in.

If no user data source is configured with ToLiMa will only user id, and name be dynamic for each user. OU will be the same as the department configured by the Token Administrator and the rest of the DN will be statically configured in global.properties.

Disabling the user data source functionality is done by commenting out the setting `ejbcaws.userdatasourcenames` in global.properties.

3.5 Customized Publisher (Optional)

ToLiMa have the possibility to reactivate ordinary cards after a temporary have been issued. In order to do this must a custom publisher have been developed that republishes the certificate that should be used to the organizations authorization systems.

For more details of how to develop a custom publisher see http://www.ejbca.org/manual.html#Custom%20publishers

Disabling the reactivation functionality is done by setting `activatecard.usereactivation` to 'false' in global.properties.

4 Installing and configuring the server

After all the initial steps have been prepared and you have all the necessary components you can start configure the server part of the framework.

4.1 Configuring HTMF

Configuration of the HMTF framework is done in several steps, and can be quite complex to install since it is dependant on several components, especially if User Data Source and Publisher plug-ins are used.

The first step is to configure the HTMF build process then the global properties of the framework. When this is done it is ready to be deployed to the application server. Next step is to reconfigure EJBCA to support the ToLiMa settings you have made, such as creating the necessary profiles and so on. Finally its time to prepare the workstation client so it can access the application.

4.1.1 The hardtokenmgmt.properties

First we edit the hardtokenmgmt build configuration. You do this by copying the hardtokenmgmt.properties.sample to hardtokenmgmt.properties and open it up in an editor. The table below give explanations to some of the most important settings.

Property	*Comment*	*Default Value*
`applet.hostname`	Used in java.policy files to allow the hardtokenmgmt applet to access the users local resources, also used to direct the EJBCA WebService traffic.	`localhost`
`applet.title`	The title in the web page displaying the applet.	`Welcome to the Hard Token Management Application`
`jarsigner.keystore`	Path to the key store used to sign all the jars	`jarsigner.jks`
`jarsigner.passphrase`	Pass phrase to unlock the store	`foo123`
`jarsigner.alias`	The alias of the key used for signing.	`jarsigner`
`customproject.location`	Path to the custom project source (optional)	`No default value`
`installpkg.appname`	Name of the application in the windows start menu	`Hard Token Management`

Property	*Comment*	*Default Value*
`installpkg.startmenu`	Defines the location of the start menu (under programs)	`Hard Token Management`
`installpkg.installall`	Include uninstall and documentation links	`true`

4.1.2 Configuring the Global Settings

In the directory src/resource/globalsettings exists a property file global.properties that it used to configure the behaviour of the HTMF application. See the developers handbook or the actual file itself for explanations about the various settings available.

Some important ToLiMa specific settings not discussed in the developers manual is described in the table below:

Property	*Comment*	*Default Value*
`token.initialbasicpin`	The default PIN used if the setting `creatingcardcontroller.generaterandompin` is set to false. The same goes for the setting `token.initialsignaturepin`.	6633
`cert.checkvalidtimeindays`	Setting indicating if the valid time should be checked in days or in percentage of the time left. If set to true it will use the `cert.validtimethreshold` in days. if false in percent.	true
`cert.validtimethreshold`	Number of days or percent of time left	10
`ejbcaws.userdatasourcenames`	`Setting defining the user data sources that should be used. This should be a ',' string containing the user data source names. If this setting is left empty or commented will the user data source functionality be disabled.`	
`adminconfigcontroller.departments`	Available departments that the token administrator can choose from in his configuration. Should be a ',' separated string.	
`errorcontroller.alwayssendreport`	Set to true if automatic reporting should be used. Then is always an e-mail sent with the stack trace file to the	False

Property	*Comment*	*Default Value*
	administrators.	
errorcontroller.promptforreportsending	Set to true if the administrator should have an option to send an error report to the system operators. Only valid if errorcontroller.alwayssendreport is set to FALSE If both 'alwayssendreport' and 'promptforreportsending' is false isn't any error reporting done.	True
errorcontroller.systemoperatormail	The mail address of were to send the error reports	
errorcontroller.smtphost	The host of were to send the reports.	localhost
errorcontroller.systemoperatorsubject	The subject of the mail	
errorcontroller.systemoperatormessage	The message sent to the system administrators before the stack trace	
activatecard.usereactivation	If this setting is set to false will the reactivate functionality be removed from the menus.	true
creatingcardcontroller.basedn	Setting defining the base DN used for users that couldn't be looked up in the user data source. To this DN will the CN and SN be added so don't put it in here. This is a required setting.	
creatingcardcontroller.defaultentityprofile	Setting defining the name of the default end entity profile used for users that couldn't be looked up in the user data source. This is a required setting.	
creatingcardcontroller.upndomain	Setting defining the domain name of a UPN (Active Directory account name) used for users that couldn't be looked up in the user data source. Comment out this setting to not set any UPN for newly created users. If set the full account name will be <user serial number>@<upndomain>	
creatingcardcontroller.canames	Setting defining the number of certificates and the CA name of those certificates in EJBCA, should be a ',' separated string of CA names. Example EIDCA1,SIGNCA1	

Property	*Comment*	*Default Value*
	if two certificate should be created. This is a required setting.	
`creatingcardcontroller.certprofilenames`	Setting defining name of the certificate profiles for each certificate placed on the card. The certificate profile name is mapped against the CA name in the setting above. I.e. the first profile is issued for the first CA name. Should be a ',' separated string of CA names. Example EIDCERT,SIGNCERT	
`creatingcardcontroller.keytypes`	Setting defining which key that should sign the pkcs10 request, ordered in the same way as the settings above. supported values are 'basic', 'sign' Example: sign,basic	
`creatingcardcontroller.certlabels`	Certificate labels on the card of the generated certificates Example: eID sign,eID auth + enc	
`creatingcardcontroller.generaterandompin`	Set to true if random PINs should be generated, otherwise will the initial PIN be used	False
`creatingcardcontroller.appendorganizationunit`	Set to true if the configured department of the administrator should be appended to the DN for later simplified search of the card	True

Important, the PKCS11 path configuration in the global.properties should refer to the path to the client and not on the server. The server doesn't need any PKCS11 installed locally.

4.2 Deploying HMTF to EJBCA

To deploy the HMTF first make sure that the EJBCA_HOME and JBOSS_HOME environment variables is set and that the Ant build tool is installed correctly . After this have been done, all you need to do is to issue the command '**ant deploy**' and the application will compile the source, a WAR application will be created containing signed jar files and it will be deployed to the application server.

4.3 Configuring EJBCA

The next step is to configure EJBCA to support the settings made in the global configuration.

4.3.1 During build time

During build time of EJBCA there is one important setting that need to be changed for ToLiMa to run. The setting is configured in the file EJBCA_HOME/con/jaxws.properties and is called `jaxws.noauthonfetchuserdata`. It should be changed from default false to true.

If you already have a instance of EJBCA up and running you have to rebuild and redeploy the application.

4.3.2 In the Web GUI

Before you begin configuring the EJBCA web GUI it's recommended that you look through the administration tutorials of EJBCA at http://docs.primekey.se/documentation/en/main-documentation.html since it demonstrates the basic concepts of that application.

How to do the actual configuration isn't covered here, just some useful tips and the requirements of ToLiMa to run.

4.3.2.1 Create the Activation Publisher (Optional)

If you want to use the functionality to activate the cards in your authorization systems automatically upon card creation you need to register the custom implemented publisher in EJBCA in the 'Edit Publisher´ page.

4.3.2.2 Create the User Data Source (Optional)

Then do a similar thing for the plug-in used to fetch user data from some existing user data store. This is done in the 'Edit User Data Sources' page. The configuration of a User Data Source is very similar to a publisher.

Remember to set to property `ejbcaws.userdatasourcenames` in the HTMF global.properties to the name of the created User Data Source. It is possible to use more that one user data source if it is necessary.

4.3.2.3 Create the CAs used by ToLiMa

The CA names configured in the property `creatingcardcontroller.canames` must exist in EJBCA.

4.3.2.4 Create the Certificate Profiles

The same goes for the certificate profiles. The names `creatingcardcontroller.certprofilenames` defined in must exist.

Make sure to uncheck the flag 'Allow validity override' and in the certificate profile that should be issue certificates sent to the authorization system, select the newly created custom publisher.

4.3.2.5 Create the End Entity Profile

The End Entity Profile defined in `creatingcardcontroller.defaultententityprofile` must also be created. Make sure it have all the certificate profiles and CAs marked as available and that the checkbox for 'Batch Generation' is checked. You also need to add the DN fields CN and SN and all the other fields that may come from `creatingcardcontroller.basedn` or the custom User Data Source. If you are generated a Microsoft Logon Certificate don't forget the UPN subject alternative name field.

4.3.2.6 Administrative Privileges

The final thing to configure in EJBCA is to set up the token administrators.

First create a administrator group and choose the CA you are going to issue the administrator cards with.

For access rules select 'Advanced Mode' and add the following rules:
TODO

Remember to set the administrator flag in the end entity data of the users that are going to be administrators. If you have problems setting up administrators check http://docs.primekey.se/documentation/en/appendixes/adminprives.html for useful tips.

4.3.2.7 System Configuration

The next thing to configure is the System Configuration, Hard Token Data encryption should be turned on and approval notifications should be configured. See the image below how these settings should be configured.

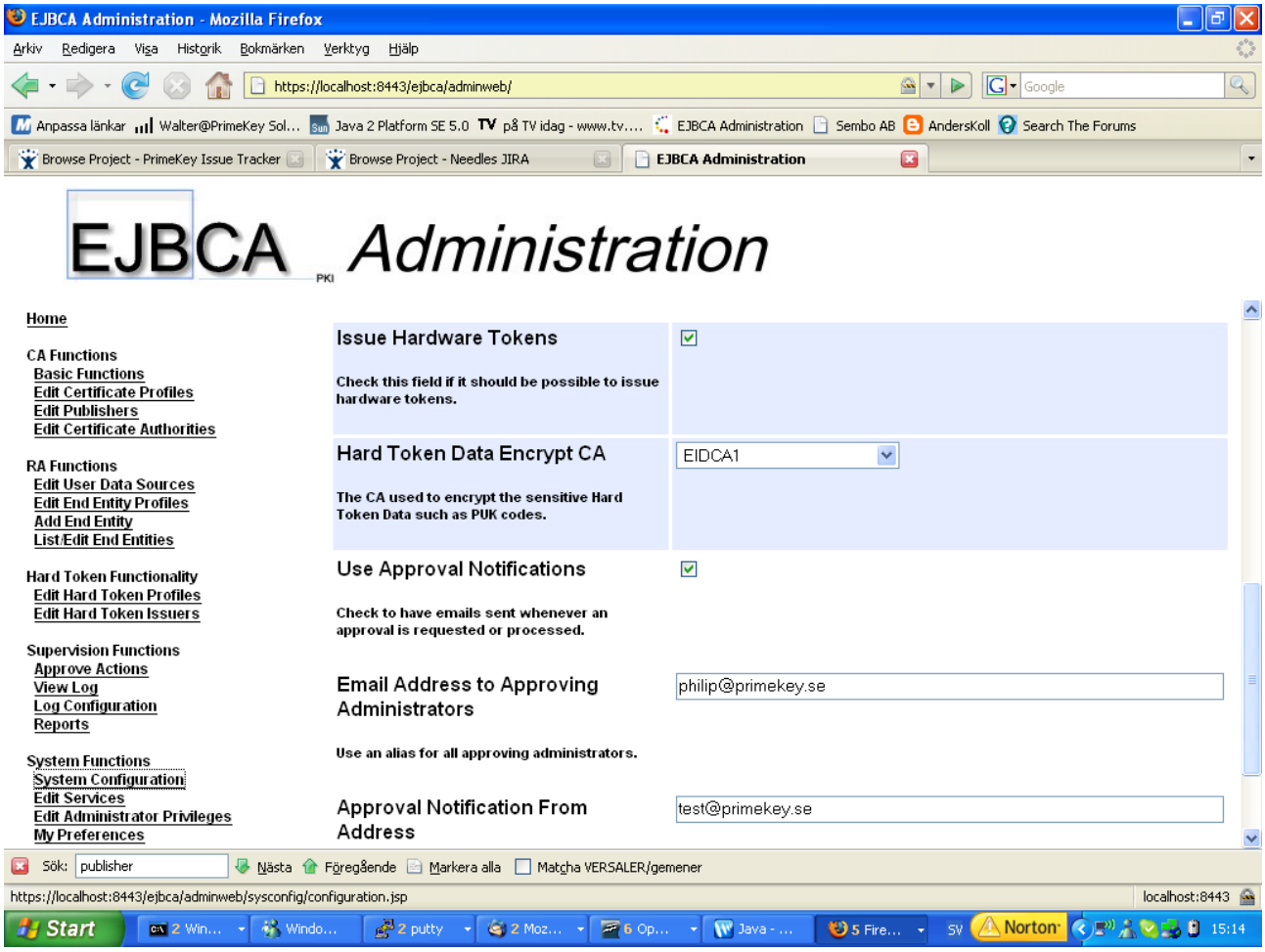

Illustration 1: System Configuration of ToLiMa

4.3.3 Import Hard Token Data

Some of the Tokens (for instance the SecCos Instant Eid cards) are pre-formatted and cannot be reformatted during the initialization phase. This means that the PUK cannot be generated but is delivered statically with the card. If this is the case make sure you get the PUK data from the supplier in electronic form so you can convert the data to a form that the EJBCA CLI import tool can parse.

The EJBCA CLI hard token data import tool is configured with the file EJBCA_HOME/src/cli/standardfilehardtokenimporter_example.properties. Copy it and and edit it's setting.

The contents of the file can be viewed here:

```
# Demonstration configuration file of a hard token importer

#property pointing the the importer that should be used.
importer.classpath=org.ejbca.ui.cli.hardtoken.importer.StandardFileHardTokenImp
orter

#Property containing the issuer DN that the tokens should be connected to
#for authorization purposes.
significantissuerdn=CN=AdminCA1,O=EJBCA Sample,C=SE

# property pointing to the file containing the data
file=test.txt

# separator of columns
separator=,

# This example indicate that the order of the columns are first token SN
#then basic pin, then signature pin and finally PUK same for both pin.
columnorder=tokensn,pin1,pin2,bothpuk

# Should be either enhancedeid or swedisheid
tokentype=swedisheid
```

The file is quite self-explaining but the most important settings are file which should point to the location where the actual token data is. The other is the column order, the available options are `'tokensn', 'pin1', 'pin2', 'bothpin', 'puk1', 'puk2', 'bothpuk',` where the token serial number refers to the complete serial number given to the PKCS11 through the token info. This isn't always the same as the one given from the token supplier.

If you have to token data in some other form like a database it is possible to import them using a plug-in. This is done by creating a class implementing `org.ejbca.ui.cli.hardtoken.importer.IHardTokenImporter` and refer to it in the importer.classpath setting. Also remember that the implemented plug-in should be in the CLI class path.

The next step is to perform the actual import using the CLI. The command is run from EJBCA_HOME and have the syntax:

```
bin\ejbca.sh/.cmd hardtoken importdata <propertyfile> -force
```

The first argument is the path to your configuration file and the -force flag indicates that token serial numbers that already exists should be overwritten.

Remember, this step is not necessary for cards that can be reformatted by the PKCS11 or token implementation.

4.4 Finally

Reboot the application server to make sure all components have the latest code and configuration.

5 Installing and configuring the client workstation

After configuring the server the final thing to do is to configure a client workstation.

5.1 Required software and hardware

The client workstation needs to have the following:

- Windows XP and above with Internet Explorer 6.0 and above
- The two card readers
- JDK 1.5 and above (http://java.sun.com)
- The PKCS11 installed on the system.
- The client install package generated by the HTMF

Currently is IE the only browser supported, this is because of the administrator authentication requirement,that the same certificate used to authenticate to the browser should be used for the applets Web Service calls to EJBCA. IE seems to be the only browser functioning with Java applets this way.

5.2 Generating the client installer

HTMF uses the NSIS tools to generate a small Windows installer containing the IAIK pkcs11 wrapper and a policy file appended to the Java policy allowing HTMF to access local resources on the computer.

The installer cannot be distributed binary because the host name of the server must be in the policy file.

The installer is generated by the command '**ant gen.win.install.pkg'** and the package is placed in the 'dist' directory. It is possible to install the client using either standard manual installation or silently from example group policy or script by executing the installer the with the '/S' parameter.

5.3 The first administration card

Before we can start issue cards with ToLiMa we have a chicken and the egg situation were you must have an administrator card to be able to issue an administrator card. To solve this you should add an administrator as in EJBCA with the token type 'User Generated' and download the certificate using the EJBCA public pages. Make sure to select a CSP that your token supports. Then make sure you have added this user to the token administrator group before you log in.

6 Starting up the application

When all this is done it should be possible to start the application by opening up the browser and accessing the URL https://<hostname>:8443/ejbca/hardtokenmgmt.

You are now all set to go.

7 More Information

More information about the Hard Token Management Framework and ToLiMa can be found at:

Main Website, http://www.hardtokenmgmt.org

EJBCA.org, http://www.ejbca.org