****

Trust Translation Authority quick guide

This document is issued within the frame and for the purpose of the LIGHT*est* project. LIGHT*est* has received funding from the European Union’s Horizon 2020 research and innovation programme under G.A. No 700321.

This document and its content are the property of the *Lightest* Consortium. All rights relevant to this document are determined by the applicable laws. Access to this document does not grant any right or license on the document or its contents. This document or its contents are not to be used or treated in any manner inconsistent with the rights or interests of the *Lightest* Consortium or the Partners detriment and are not to be disclosed externally without prior written consent from the *Lightest* Partners.

Each *Lightest* Partner may use this document in conformity with the *Lightest* Consortium Grant Agreement provisions.

**NOT TO BE DISTRIBUTED OUTSIDE THE LIGHTEST CONSORTIUM**

# Document Information

## Contributors

|  |  |
| --- | --- |
| **Name** | **Partner** |
| Miguel Angel Mateo | Atos |

# Table of Contents

[1. Document Information 2](#_Toc30505539)

[1.1 Contributors 2](#_Toc30505540)

[2. Table of Contents 3](#_Toc30505541)

[3. REST API Description 4](#_Toc30505542)

[3.1 Create an agreement of Translation 4](#_Toc30505543)

[3.2 Retrieve names of available agreements of translation 4](#_Toc30505544)

[3.3 Retrieve details of an agreement of translation 4](#_Toc30505545)

[3.4 Delete an agreement of Translation 5](#_Toc30505546)

[3.5 Get information about agreements by Trust Scheme 5](#_Toc30505547)

[3.6 Get Trust Schemes 5](#_Toc30505548)

[3.7 Get Trust Scheme details 6](#_Toc30505549)

[4. Configuration of TTA: 8](#_Toc30505550)

[4.1 List of parameter: 8](#_Toc30505551)

[4.2 Configuration REST API 10](#_Toc30505552)

[4.3 Set configuration parameter 12](#_Toc30505553)

[5. TTA file downlading service : 13](#_Toc30505554)

[6. EXAMPLE OF DEPLOYMENT 14](#_Toc30505555)

[6.1 SSL Certificates: 14](#_Toc30505556)

[6.2 Application servers 14](#_Toc30505557)

[6.3 Signing certificates 14](#_Toc30505558)

[6.4 TTA service compiling 14](#_Toc30505559)

[6.5 TTA deployment and testing 15](#_Toc30505560)

# REST API Description

## Create an agreement of Translation

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Name | Resource | Method | Body | Description |
| Create translation | /translation | POST | See body example 1 | Create an agreement of translation |
| Example | POST <https://tta-lightest.eu:8443/integration/ttaFM/mng/rsc/translation> | | | |
| Answer | 200 OK | {  "agreename":"eidas-esig\_fido",  "xmlFile":"fido.xml",  "tplFile":"fido.tpl.p7s",  } | | |

Body example:

{

"agreement": {

"name": "eidas-esig\_fido",

"status": "active",

"creation-date": "2019-08-10",

"leaving-date": "2020-08-10",

"activation-date": "2019-08-10",

"source": {

"name": "eidas-esignature",

"level": "advanced",

"provider": "eu"

},

"target": {

"name": "fido",

"provider": "usa",

"params": [{

"name": "param1",

"value": "value1"

},

{

"name": "param2",

"value": "value2"

}

]

}

}

}

Description:

Agreement->name: (mandatory) name of the translation agreement

Agreement->status: (mandatory) internal use, set to “active”.

Agreement-> \*-date: (optional) format YYYY-MM-DD

Agreement->source: (Mandatory) Who provider trust, origin of trust.

Agreement->target: (Mandatory) Who is trusted as result of the agreement.

Source, Target:

name: (Mandatory) name of a trust scheme.

level: (Optional) level of a trust shceme

provider: (Optional)

params: (optional) List of params

params->name: name of param

params->value: value of param

## Retrieve names of available agreements of translation

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Name | Resource | Method | Body | Description |
| get translations | /translation | GET |  | Retrieve a listo f names of all available agreements of translation |
| Example | GET <https://tta-lightest.eu:8443/integration/ttaFM/mng/rsc/translation> | | | |
| Answer | 200 OK | {  "agreementNames": [  "eidas-esig\_fido"  ]  } | | |

## Retrieve details of an agreement of translation

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Name | Resource | Method | Body | Description |
| get agreement | /translation/<name> | GET |  | Retrieve details of an agreement of translation |
| Example | GET https://tta-lightest.eu:8443/unittesting/ttaFM/mng/rsc/translation/eidas-esig\_fido | | | |
| Answer | 200 OK | The body of the answer is the one used to create the agreement of translation. See Body example 1 | | |

## Delete an agreement of Translation

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Name | Resource | Method | Body | Description |
| Delete translation | /translation/<name> | DELETE |  | Delete an agreements of translation |
| Example | DELETE https://tta-lightest.eu:8443/unittesting/ttaFM/mng/rsc/translation/eidas-esig\_fido | | | |
| Answer | 200 OK |  | | |

## Get information about agreements by Trust Scheme

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Name | Resource | Method | Body | Description |
| Get agreements by trust scheme | / getAgreementsRelatedToATrustScheme/<trust sheme name> | GET |  | It answer a list of agreement names in which the given trust scheme participates as source and as origin |
| Example | GET https://tta-lightest.eu:8443/unittesting/ttaFM/mng/rsc/getAgreementsRelatedToATrustScheme/fido | | | |
| Answer | 200 OK | {  "trustSchemeName": "fido",  "TargetOfTrustIn": [  "eidas-esig\_fido"  ],  "OriginOfTrustIn": []  } | | |

## Get Trust Schemes

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Name | Resource | Method | Body | Description |
| Get trust scheme details | /trustScheme | GET |  | Get a list of available Trust Schemes |
| Example | GET https://tta-lightest.eu:8443/unittesting/ttaFM/mng/rsc/trustScheme | | | |
| Answer | 200 OK | {  "trustSchemeNames": [  "fido",  "eidas-esignature"  ]  } | | |

## Get Trust Scheme details

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Name | Resource | Method | Body | Description |
| Get trust scheme details | /trustScheme/<TSname> | GET |  | Get details of a Trust Scheme |
| Example | GET https://tta-lightest.eu:8443/unittesting/ttaFM/mng/rsc/trustScheme/fido | | | |
| Answer | 200 OK | {  "trust-scheme-list": [  {  "provider": "usa",  "name": "fido",  "params": [  {  "param1": "value1"  },  {  "param2": "value2"  }  ]  }  ]  } | | |

# Comments about how TTA manage Trust Schemes

TTA differentiate Trust Schemes by the value of the “name” field. Then, a Trust Scheme can have different definitions depending of the value of all its properties.

This means that there should be a commitment regarding how Trust Schemes are named. For instance, in the case of Eidas, we can set as TS name EIDAS\_<service>, so TTA will manage EIDAS\_eSignature and EIDAS\_eSeal, the information retrieved with the trustscheme resource API will be:

GET <https://tta-lightest.eu:8443/unittesting/ttaFM/mng/rsc/trustScheme>

{

"trustSchemeNames": [

" eidas-eseal ",

"eidas-esignature"

]

}

GET <https://tta-lightest.eu:8443/unittesting/ttaFM/mng/rsc/trustScheme/eidas-eseal>

{

"trust-scheme-list": [

{

"level": "advanced",

"provider": "eu",

"name": "eidas-eseal"

}

]

}

GET <https://tta-lightest.eu:8443/unittesting/ttaFM/mng/rsc/trustScheme/eidas-esignature>

{

"trust-scheme-list": [

{

"level": "quelified",

"provider": "eu",

"name": "eidas-esignature"

}

]

}

Other alternative is to set EIDAS as the name of the Trust Scheme adding the service name as a param; in this case TTA will manage EIDAS as a Trust Scheme with two definitions.

GET <https://tta-lightest.eu:8443/unittesting/ttaFM/mng/rsc/trustScheme>

{

"trustSchemeNames": [

" eidas"

]

}

GET <https://tta-lightest.eu:8443/unittesting/ttaFM/mng/rsc/trustScheme/eidas>

{

"trust-scheme-list": [

{

"name": "eidas",

"provider":"eu",

"params": [

{

"level": "quelified"

},

{

"service": "esignature"

}

]

},

{

"name": "eidas",

"provider":"eu",

"params": [

{

"level": "advanced"

},

{

"service": "eseal"

}

]

}

]

}

This behavior is especially useful to manage Tuple base Trust Schemes, where a Trust Scheme can have several definitions depending on attributes and their values.

According to deliverables, TTA considers that the name of a trust scheme is formed in the way: TrustSchemeName\_Service\_Level, although the service also lets to set name of a Trust Scheme as: TrustSchemeName\_Service, adding level, if available, as a parameter

There should be also considered that removing an agreement of translation does not result in removing the definition of Trust Schemes, even when a specific trust scheme does not participate, any more, in any agreement of translation.

# Configuration of TTA

## List of parameter:

* db\_installed: (DO NOT MODIFY)

value: true, false

description: TTA module keep all information about agreements of translations into internal storage in order to provide fault tolerance. When this parameter is set to true the storage is provided by a MongoDB, When this parameter is set to false the storage is provided by the file system.

* *db\_url*: (DO NOT MODIFY)
  + value: IP address
* *db\_port*: (DO NOT MODIFY)
  + value: port number
* *mongo-db-user*: (DO NOT MODIFY)
  + value: user name
* *mongo-db-pwd*: (DO NOT MODIFY)
  + value: password
* *mongo-db-name*: (DO NOT MODIFY)
  + value: name of the data base
* *agreement-collection*: (DO NOT MODIFY)
  + value: name of the collection to store agreements
* *scheme-collection*: (DO NOT MODIFY)
  + value: name of the collection to store trust schemes
* *file-storage-dir*: (DO NOT MODIFY)
  + value: path of the directory for the file system storate
* *sign*: Indicates if Trust Translation declaration files has to be signed or no. In the case they are signed the public key of the certificate is published in SMIMEA resource record.
  + value: true, false
* *pkcsfile*: (DO NOT MODIFY)path/name of the container file used for the signing certificate (PKCS12 format)
  + value: absolute file name
* *pkcspwd*: (DO NOT MODIFY) password of the file used for the signing certificate (PKCS12 format)
  + value: alphanumerical string

Configuration parameter to sign XML files:

* *xml\_digest\_algorithm*:
  + value: SHA1, SHA224, SHA256, SHA384, SHA512
* *xml\_token\_type*: (DO NOT MODIFY)
  + value: PKCS12
* *xml\_signature\_packaging*:
  + value: ENVELOPED, ENVELOPING, DETACHED (this last one, although supported by signing mechanism is not consistent with file server mechanism)
* *xml\_signature\_level type:* 
  + value: CAdES\_BASELINE\_LTA, CAdES\_BASELINE\_LT, CAdES\_BASELINE\_T, CAdES\_BASELINE\_B
* *tpl\_digest\_algorithm* 
  + value: SHA1, SHA224, SHA256, SHA384, SHA512
* *tpl\_token\_type*: (DO NOT MODIFY)
  + value: PKCS12
* *tpl\_signature\_packaging* 
  + value: ENVELOPING, DETACHED (this last one, although supported by signing mechanism is not consistent with file server mechanism)
* *tpl\_signature\_level* 
  + value: CAdES\_BASELINE\_LTA, CAdES\_BASELINE\_LT, CAdES\_BASELINE\_T, CAdES\_BASELINE\_B
* *dnsConfigured*: Indicates if there is a DNS available and configured. TESTING purposes only to avoid sending unuseful request to DNS.
  + value: true, false
* *dns-domain*:
  + value: Domain of the DNS to be added to the trust files names for the DNS API
* *dnsURL*:
  + value: URL of the REST API of the DNS zone manager.
* *token\_file*: (DO NOT MODIFY)
  + value: absolute name of the file containing the authentication token to connect to Zone Manger
* *fileServerURL*: (DO NOT MODIFY)
  + value: URL base of the TTA file server to download trust translation files
* *file-server-dir*: (DO NOT MODIFY)
  + value: file system pat where Trust Translation Files
* *one\_file\_per\_scheme*: This parameter indicates whether trust translation files are built for Transtlation files of for Translation Agreements.

When true TTA build one file (one TPL, one XML) per Trust Scheme, describing all agreements in which it participates.

When false TTA build one file (one TPL, one XML) per Translation agreement.

* + value: true, false
* *dns\_uri\_record\_target*: When Trust Translation files are built for Agreements (one\_file\_per\_scheme = false) this parameter indicates whether the URI Resource record is build with the name of the Trust Scheme of the source or the name of the Trust Scheme of the target.
  + value: true, false

**Consider that the behavior described in deliverables is that TTA produces a file per Agreement, and that the URI for the Resource Record is built with the name of the target, this mean that there will be several RR in DNS with the same Trust Scheme name pointing to different translation files**.

\_translation.\_trust.fido.lightest.nlnetlabs.nl. 3599 IN URI 10 1 "https://integration.tta-lightest.eu/ttaFM/mng/TrustTranslationDeclaration/eidas\_esignature-fido"

\_translation.\_trust.fido.lightest.nlnetlabs.nl. 3599 IN URI 10 1 “<https://integration.tta-lightest.eu/ttaFM/mng/TrustTranslationDeclaration/eidas_eseal-fido>”

**This behavior is not supported by the Zone Manager version used, so the configuration should be:**

one\_file\_per\_scheme = TRUE

Also there could be incongruences if you change this parameters without removing all agreements previously created.

# Configuration REST API

## Get configuration

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Name | Resource | Method | Body | Description |
| Get configuration | / cfg/conf | GET |  | Get a configuration of TTA |
| Example | https://tta-lightest.eu:8443/unittesting/ttaFM/mng/cfg/conf | | | |
| Answer | 200 OK | {  "tta-configuration": [  {  "xml\_digest\_algorithm": "SHA512"  },  {  "db\_url": "172.24.76.37"  },  {  "db\_port": "25555"  },  {  "one\_file\_per\_scheme": "true"  },  {  "db\_installed": "false"  },  {  "xml\_signature\_level": "XAdES\_BASELINE\_LT"  },  {  "token\_file": "../lightest/dnsToken"  },  {  "file-storage-dir": "../lightest/fileStorage"  },  {  "tpl\_signature\_packaging": "ENVELOPING"  },  {  "mongo-db-pwd": "60"  },  {  "tpl\_digest\_algorithm": "SHA512"  },  {  "pkcspwd": "xxx"  },  {  "pkcsfile": "../lightest/tta.pfx"  },  {  "dnsURL": "https://lightest.tta.es/names"  },  {  "xml\_token\_type": "PKCS12"  },  {  "scheme-collection": "schemes"  },  {  "tpl\_token\_type": "PKCS12"  },  {  "xml\_signature\_packaging": "ENVELOPED"  },  {  "mongo-db-user": "dockerUser"  },  {  "mongo-db-name": "lightest"  },  {  "fileServerURL": "https://tta-lightest.eu:8443/unittesting/ttaFM/mng/TrustTranslationDeclaration"  },  {  "dns-domain": ".lightest.nlnetlabs.nl"  },  {  "tpl\_signature\_level": "CAdES\_BASELINE\_LT"  },  {  "agreement-collection": "agreements"  },  {  "file-server-dir": "../lightest/tta/files/"  },  {  "dnsConfigured": "true"  },  {  "sign": "true"  },  {  "dns\_uri\_record\_target": "true"  }  ]  } | | |

## Set configuration parameter

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Name | Resource | Method | Body | Description |
| Set param | /trustScheme | PUT | {"name": "sign",  "value": "false"  } | Set value of a configuration parameter |
| Example | https://tta-lightest.eu:8443/unittesting/ttaFM/mng/cfg/param | | | |
| Answer | 200 OK |  | | |

# TTA file downlading service

TTA works as File Server to store Trust Translation files, and provides the corresponding file downlading service

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Name | Resource | Method | Body | Description |
| GET TTA file | /TrustTranslationDeclaration/<trust\_scheme\_name> | GET |  | Download TT declaration file |
| Example | https://tta-lightest.eu:8443/unittesting/ttaFM/mng/TrustTranslationDeclaration/fido | | | |
| Answer | 200 OK |  | | |

This service take into account HTTP header “accept”

If Accept header contains \*xml\* the file provided will be xml trust translation file, any other case the file provided will be TPL format.

# Example of Deployment

This is an example of how to deploy the TTA module in a Apache Tomcat server with TSL. We encourage to take a look to the documentation of the application server used for the deployment.

## SSL Certificates:

To obtain valid X.509 certificate for the domain assigned to the server, in this case we will use the domain: tta-lightest.eu

:~/tomcat/keystore/cert$ ll

total 32

drwxrwxr-x 2 admin 4096 May 13 13:41 ./

drwxr-xr-x 3 admin 4096 May 13 12:46 ../

-rw-rw-r-- 1 admin 1915 May 10 12:51 cert3.pem

-rw-rw-r-- 1 admin 1647 May 10 12:51 chain3.pem

-rw-rw-r-- 1 admin 3562 May 10 12:51 fullchain3.pem

-rw-rw-r-- 1 admin 1704 May 10 12:51 privkey3.pem

## Application servers

To deploy Apache Tomcat server, in this example we use Tomcat v8: <https://tomcat.apache.org/download-80.cgi>

Set the https connector by editing its configuration file server.xml and setting the certificates for TSL:

<Connector port="8081" protocol="HTTP/1.1" connectionTimeout="20000" redirectPort="8441" />

<Connector port="8441" protocol="org.apache.coyote.http11.Http11NioProtocol" maxThreads="150" SSLEnabled="true" >

<SSLHostConfig>

<Certificate certificateKeyFile="/home/tomcat/keystore/cert/privkey3.pem"

certificateFile="/home/tomcat/keystore/cert/cert3.pem"

certificateChainFile="/home/tomcat/keystore/cert/chain3.pem" />

</SSLHostConfig>

</Connector>

<Connector port="8009" protocol="AJP/1.3" redirectPort="8441" />

## Signing certificates

To create the certificate container to be used in for the signing of documents. In this case, and only for testing purposes, we use same certificate that the one used for SSL:

miguel.mateo@LIGHTEST:~/tomcat/keystore/cert$ openssl pkcs12 -export -out tta.pfx -inkey privkey3.pem -in cert3.pem -certfile fullchain3.pem

Enter Export Password:

Verifying - Enter Export Password:

admin@LIGHTEST:~/tomcat/keystore/cert$ ll

total 32

drwxrwxr-x 2 miguel.mateo miguel.mateo 4096 May 13 13:41 ./

drwxr-xr-x 3 miguel.mateo miguel.mateo 4096 May 13 12:46 ../

-rw-rw-r-- 1 miguel.mateo miguel.mateo 1915 May 10 12:51 cert3.pem

-rw-rw-r-- 1 miguel.mateo miguel.mateo 1647 May 10 12:51 chain3.pem

-rw-rw-r-- 1 miguel.mateo miguel.mateo 3562 May 10 12:51 fullchain3.pem

-rw-rw-r-- 1 miguel.mateo miguel.mateo 1704 May 10 12:51 privkey3.pem

-rw-rw-r-- 1 miguel.mateo miguel.mateo 5613 May 13 13:42 tta.pfx

The resulting file “tta.pfx” is the one that should to be set as value of the configuration property: *pkcsfile*

Example: /Home/tomcat/keystore/tta.pfx

The password used to create this file have to set as value of the configuration property: *pkcspwd*

## TTA service compiling

Clone and build TTA java project to be deployed in the Tomcat server.

Note. Be sure you have installed Java 8 and git and maven in your server.

Clone TTA project:

$ git clone <https://extgit.iaik.tugraz.at/LIGHTest/TTA.git>

Cloning into 'TTA'...

Username for 'https://extgit.iaik.tugraz.at': xxx

Password for xxx:

remote: Enumerating objects: 479, done.

remote: Counting objects: 100% (479/479), done.

remote: Compressing objects: 100% (225/225), done.

remote: Total 479 (delta 206), reused 383 (delta 160)

Receiving objects: 100% (479/479), 175.09 KiB | 3.24 MiB/s, done.

Resolving deltas: 100% (206/206), done.

admin@lightest:~/ttt$

Navigate to TTA and take a look to files.

$ cd TTA/

$ ll

total 362

-rw-r--r-- 1 admin 1049089 feb 6 2019 LICENSE.txt

-rw-r--r-- 1 admin 1049089 abr 17 2019 pom.xml

-rw-r--r-- 1 admin 1049089 ene 21 14:55 Readme.md

-rw-r--r-- 1 admin 1049089 feb 6 2019 rest\_log4j.properties

drwxr-xr-x 1 admin 1049089 abr 17 2019 src/

drwxr-xr-x 1 admin 1049089 jun 14 2019 target/

-rw-r--r-- 1 admin 1049089 ene 21 14:40 TTA\_service\_quick\_guide.docx

-rw-r--r-- 1 admin 1049089 ene 21 15:03 ttaFM.properties

Compile the project:

$ mvn clean install

…

[INFO] ------------------------------------------------------------------------

[INFO] BUILD SUCCESS

[INFO] ------------------------------------------------------------------------

[INFO] Total time: 5.790 s

[INFO] Finished at: 2020-01-21T10:26:47Z

[INFO] ------------------------------------------------------------------------

miguel.mateo@lightest:~/ttt/TTA$

## TTA deployment and testing

Copy the package war file created in the previous step to the deployment directory of the Tomcat server.

$ cp ./target/ttaFM.war /home/tomcat/webapps

Start the server:

$ cd /home/tomcat/bin

./startup.sh

$ ./startup.sh

Using CATALINA\_BASE: /home/tomcat/

Using CATALINA\_HOME: /home/tomcat/

Using CATALINA\_TMPDIR: /home/tomcat/temp

Using JRE\_HOME: /usr/lib/jvm/java-8-openjdk-amd64/

Using CLASSPATH: /home/ /tomcat/bin/bootstrap.jar:/home/tomcat/bin/tomcat-juli.jar

Tomcat started.

Copy configuration files for TTA and logs to Tomcat server

$ cp ttaFM.properties /home/tomcat/conf

$ cp rest\_log4j.properties /home/tomcat/conf

and test in any web explorer navegating to the url:

<https://tta-lightest.eu:8441/ttaFM/mng/rsc/translation/>

Note. Be aware of the port used, in this case we set port 8441 for TSL, and this is the one set in the url.

You should get the following answer from TTA in the navigator:

{"agreementNames":[]}

The output means that the TTA is up and running and that it does not have any Translation agreement yet.